(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2019년02월27일
(11) 등록번호 10-1952360
(24) 등록일자 2019년02월20일

(51) 국제특허분류(Int. Cl.)
H03M 7/40 (2006.01) *H04N 19/60* (2014.01)
H04N 19/91 (2014.01)
(52) CPC특허분류
H03M 7/4018 (2013.01)
H03M 7/4075 (2013.01)
(21) 출원번호 10-2017-7023278(분할)
(22) 출원일자(국제) 2013년01월21일
심사청구일자 2018년01월18일
(85) 번역문제출일자 2017년08월21일
(65) 공개번호 10-2017-0100048
(43) 공개일자 2017년09월01일
(62) 원출원 특허 10-2014-7034588
원출원일자(국제) 2013년01월21일
심사청구일자 2014년12월09일
(86) 국제출원번호 PCT/EP2013/051053
(87) 국제공개번호 WO 2013/107908
국제공개일자 2013년07월25일
(30) 우선권주장
61/588,846 2012년01월20일 미국(US)
(56) 선행기술조사문헌
JP2011035682 A

(73) 특허권자
지이 비디오 컴프레션, 엘엘씨
미국 뉴욕 12211 올버니 사우스우즈 블러바드 8
(72) 발명자
뉘엔, 퉁
독일 13055 베를린 란츠베르거 알리 217
키르호호퍼, 하이너
독일 10555 베를린 고츠코브스키슈트라쎄 5
마르페, 데트레브
독일 12161 베를린 수에드베스트코르소 70
(74) 대리인
윤의섭, 김수진

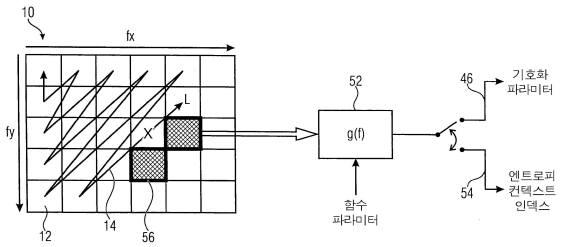
전체 청구항 수 : 총 29 항 심사관 : 조춘근

(54) 발명의 명칭 **변환 계수 코딩**

(57) 요 약

이전에 코딩된/디코딩된 변환 계수들 상에 기호화 파라미터의 의존도 및 컨텍스트의 의존도에 대해 동일한 함수를 이용하는 것이 여기에 이용된 아이디어이다. 동일 함수를 이용하는 것은 - 다양한 함수 파라미터를 변화시키는 것과 함께 - 변환 블록들에 공간적으로 배치되는 변환 계수들의 경우에 변환 블록들의 상이한 변환 블록 크기들 및/또는 주파수 부분들에 관해 이용될 수도 있다. 현재 변환 계수가 변환 블록 내에 위치하며, 현재 변환 계수 블록의 상이한 크기들, 현재 변환 계수의 변환 블록의 상이한 정보 구성요소 타입들 및/또는 상이한 주파수 부분들에 대한 이전에 코딩된/디코딩된 변환 계수들 상의 기호화 파라미터의 의존도에 대해 동일한 함수를 이용하는 것이 이 아이디어의 추가 변형례이다.

대 표 도



(52) CPC특허분류
H04N 19/60 (2015.01)
H04N 19/91 (2015.01)

명 세 서

청구범위

청구항 1

데이터 스트림(32)으로부터 변환 계수 레벨을 갖는 복수의 변환 계수들(12)을 디코딩하는 장치에 있어서,

데이터 스트림(32)으로부터 하나 이상의 기호들의 제1집합(44)을, 현재 변환 계수 (x)에 대해, 엔트로피 디코딩하도록 구성되는 컨텍스트 적응 엔트로피 디코더(80);

제1기호화 설계에 따라 제1레벨 인터벌(16) 내에서 변환 계수 레벨 상에 하나 이상의 기호들의 상기 제1집합(44)을 맵핑하도록 구성되는 역기호화기(82);

하나 이상의 기호들의 제1집합이 제1기호화 설계에 따라 맵핑되는 변환 계수 레벨이 상기 제1레벨 인터벌의 최대 레벨인 경우, 상기 데이터 스트림(32)으로부터 기호들의 제2집합(42)을 추출하도록 구성되는 추출기(84);를 포함하며,

상기 역기호화기(82)는 기호화 파라미터(46)에 따라 파라미터화 될 수 있는 제2기호화 설계에 따라 제2레벨 인터벌(18) 내에서의 위치에서 기호들의 제2집합을 맵핑하도록 구성되며,

상기 컨텍스트 적응 엔트로피 디코더(80)는, 상기 데이터 스트림(32)으로부터 하나 이상의 기호들의 상기 제1집합(44)의 적어도 하나의 미리 결정된 기호를 엔트로피 디코딩하는데 있어, 함수 파라미터를 통해 파라미터화 될 수 있는 함수(52)를 통해, 제1설정으로 상기 함수 파라미터를 설정하는 것과 함께, 이전에 디코딩된 변환 계수들에 의존하는 컨텍스트를 이용하도록 구성되며,

하나 이상의 기호들의 상기 제1집합(44)이 상기 제1기호화 설계에 따라 맵핑되는 변환 계수 레벨이 상기 제1레벨 인터벌(16)의 최대 레벨인 경우, 상기 함수 파라미터가 제2설정으로 설정되는 것과 함께 상기 함수(52)을 통해, 상기 이전에 디코딩된 변환 계수들에 의존하는 상기 기호화 파라미터(46)를 결정하도록 구성되는 기호화 파라미터 결정기(86)를 더 포함하며,

상기 복수의 변환 계수들은 픽처의 블록을 상이한 공간적 주파수들로 분해하는 변환을 표현하는 변환 계수 블록에 포함되는 것을 특징으로 하는,

데이터 스트림(32)으로부터 변환 계수 레벨을 갖는 복수의 변환 계수들(12)을 디코딩하는 장치.

청구항 2

제1항에 있어서,

상기 장치는 상기 함수가,

$g(f(\mathbf{x}))$ 및 $g(x)=\sum_{i=1}^{d_f}\delta'(x,n_i)$ 및 $f(\mathbf{x})=\sum_{i=1}^{d}w_i\cdot h\cdot\delta(x_i,t)$

와 함께

$$\delta(x,t)=\begin{cases}1 & |x|\geq t\\ 0 & |x|<t\end{cases}$$ 및 $$\delta(x,n)=\begin{cases}1 & x>n\\ 0 & x\leq n\end{cases}$$

이도록 구성되며,

여기서 t, w_i 및 $\{n_{21}, \ldots, n_{d_f}\} = n$ 는 함수 파라미터를 형성하고,

x_i 에서 $i\in\{1\ldots d\}$인 $x=\{x_1,\ldots,x_d\}$ 는 이전에 디코딩된 변환 계수 i를 나타내며,

w_i 는 각각이 일과 동일하거나 일과 동일하지 않을 수 있는 가중 값들(weighting values)이며, h는 상수 또는 x_i의 함수인 것을 특징으로 하는 장치.

청구항 3

삭제

청구항 4

제2항에 있어서,

상기 기호화 파라미터 결정기는 상기 함수를 통해 상기 이전에 디코딩된 변환 계수들에 의존하여 기호화 파라미터가 결정되며,

x_i 가 상기 이전에 디코딩된 변환 계수 i의 상기 변환 계수 레벨과 동일하고, 상기 제1 또는 제2레벨 인터벌 내에 있는 상기 이전에 디코딩된 변환 계수 i의 상기 변환 계수 레벨로부터 독립적으로 구성되는 것을 특징으로 하는 장치.

청구항 5

제2항에 있어서,

상기 장치는 $n_1 < \ldots < n_{d_i}$인 것을 특징으로 하는 장치.

청구항 6

제2항에 있어서,

상기 장치는 h 가 $|x_i| - t$ 인 것을 특징으로 하는 장치.

청구항 7

제1항에 있어서,

상기 장치는 상기 현재 변환 계수에 관련된 상기 변환 계수들의 상대적 공간적인 배치에 의존하여 상기 이전에 디코딩된 변환 계수들을 공간적으로 결정하도록 구성되는 것을 특징으로 하는 장치.

청구항 8

제1항에 있어서,

상기 장치는 상기 데이터 스트림(32)으로부터 미리 결정된 스캔 순서(14)를 따라 변환 계수 블록의 최종 비-제로(non-zero) 변환 계수의 위치에 대한 정보를 추출하도록 구성되며, 상기 복수의 변환 계수들은 상기 스캔 순서에 따른 상기 최종 비-제로 변환 계수로부터 미리 결정된 스캔 순서의 시작에 이르기까지 상기 변환 계수 블록들의 변환 계수들을 포함하는 것을 특징으로 하는 장치.

청구항 9

제8항에 있어서,

상기 컨텍스트 적응 엔트로피 디코더는 상기 최종 비-제로 변환 계수 외의 하나 이상의 기호들의 상기 제1집합을 엔트로피 디코딩하는데 이용되는 컨텍스트들과 구분된, 상기 최종 비-제로 변환 계수에 대한 하나 이상의 기호들의 상기 제1집합을 엔트로피 디코딩하기 위한 컨텍스트들의 개별 집합을 이용하도록 구성되는 것을 특징으로 하는 장치.

청구항 10

제8항에 있어서,

상기 컨텍스트 적응 엔트로피 디코더는 상기 최종 비-제로 변환 계수로부터 미리 결정된 스캔 순서의 시작까지 이끄는 반대 스캔 순서로 상기 복수의 변환 계수들을 검색하는 것을 특징으로 하는 장치.

청구항 11

제1항에 있어서,

상기 장치는 두개의 스캔(scans)으로 상기 데이터 스트림으로부터 복수의 변환 계수들을 디코딩하도록 구성되며, 상기 컨텍스트 적응 엔트로피 디코더는 상기 변환 계수들의 제1스캔에 대응하는 순서로 상기 데이터 스트림으로부터 상기 변환 계수들에 대한 기호들의 상기 제1집합을 엔트로피 디코딩하도록 구성되며, 상기 추출기는 이후에 기호들의 상기 제1집합이 상기 변환 계수들의 제2스캔 내에서 상기 제1레벨 인터벌의 최대 레벨 상에 맵핑되는 상기 변환 계수들의 발생에 대응하는 순서로 기호들의 제1집합이 상기 데이터 스트림으로부터 상기 제1레벨 인터벌의 최대 레벨 상에 맵핑되는 상기 변환 계수들에 대한 기호들의 제2집합을 추출하도록 구성되는 것을 특징으로 하는 장치.

청구항 12

제1항에 있어서,

상기 장치는 하나의 스캔에서 순차적으로 상기 데이터 스트림으로부터 복수의 변환 계수들을 디코딩하도록 구성되며, 기호들의 제2집합은 상기 변환 계수들의 기호들의 제1집합 사이에서 상기 데이터 스트림 내에 배치되며, 상기 컨텍스트 적응 엔트로피 디코더 및 추출기는, 하나의 스캔의 스캔 순서로 각 변환 계수에 대해, 기호들의 제1집합이 상기 데이터 스트림으로부터 상기 제1레벨 인터벌의 상기 최대 레벨 상에 맵핑되는 각각의 변환 계수들의 하나 이상의 기호들의 상기 제1집합의 상기 컨텍스트 적응 엔트로피 디코더의 엔트로피 디코딩 바로 다음에 기호들의 제1집합이 상기 데이터 스트림으로부터 상기 제1레벨 인터벌의 최대 레벨 상에 맵핑되는 각각의 변환 계수들의 기호들의 제2집합들을 추출하도록 구성되는 것을 특징으로 하는 장치.

청구항 13

제1항에 있어서,

상기 추출기는 고정된 개연성 분포를 이용하는 엔트로피 디코딩을 이용하거나 상기 데이터 스트림으로부터 직접 기호들의 제2집합을 추출하도록 구성되는 것을 특징으로 하는 장치.

청구항 14

제1항에 있어서,

상기 제1기호화 설계는 절단 단항 이진 설계(truncated unary binarization scheme)인 것을 특징으로 하는 장치.

청구항 15

제1항에 있어서,

상기 제2기호화 설계는 기호들의 상기 제2집합이 라이스 코드(Rice code)인 것을 특징으로 하는 장치.

청구항 16

제1항에 있어서,

상기 데이터 스트림이 깊이 맵(depth map)을 포함하는 것을 특징으로 하는 장치.

청구항 17

변환 계수 레벨들을 갖는 복수의 변환 계수들을 데이터 스트림(32)으로 코딩하는 장치에 있어서,

현재 변환 계수의 변환 계수 레벨이 제1레벨 인터벌(16) 내에 있는 경우, 제1기호화 설계에 따라 하나 이상의 기호들의 제1집합 상에, 그리고

상기 현재 변환 계수의 변환 계수 레벨이 제2레벨 인터벌(18) 내에 있는 경우, 기호화 파라미터(46)에 따라 파라미터화 가능한 제2기호화 설계에 따라, 제2레벨 인터벌(18) 내의 현재 변환 계수의 변환 계수 레벨의 위치에 의존하는 기호들의 제3집합, 그리고 제1레벨 인터벌(16)의 최대 레벨이 제1기호화 설계에 따라 맵핑되는 기호들의 제2집합의 조합(combination) 상에,

현재 변환 계수를 맵핑하도록 구성되는 기호화기(34);

만약 상기 현재 변환 계수의 변환 계수 레벨이 상기 제1레벨 인터벌 내인 경우, 하나 이상의 기호들의 상기 제1집합을 데이터 스트림으로 엔트로피 인코딩하도록, 상기 현재 변환 계수의 변환 계수 레벨이 상기 제2레벨 인터벌 내인 경우, 하나 이상의 기호들의 제2집합을 데이터 스트림으로 엔트로피 인코딩하도록 구성되며, 하나 이상의 기호들의 제2집합의 적어도 하나의 미리 결정된 기호를 데이터 스트림으로 엔트로피 인코딩하는데 있어, 제1설정으로 설정되는 함수 파라미터와 함께, 함수 파라미터를 통해 파라미터화 가능한 함수(52)를 통해, 이전에 코딩된 변환 계수들 상에, 의존하는 컨텍스트를 이용하도록 구성되는, 컨텍스트 적응 엔트로피 인코더(36);

만약 상기 현재 변환 계수의 변환 계수 레벨이 상기 제2레벨 인터벌 내인 경우, 제2설정으로 설정된 상기 함수 파라미터를 갖는 함수를 통해, 상기 이전에 코딩된 변환 계수들 상에, 의존하는 기호들의 제3집합 상에 맵핑하기 위해 상기 기호화 파라미터(46)을 결정하도록 구성되는 기호화 파라미터 결정기(38); 및

만약 상기 현재 변환 계수들의 변환 계수 레벨이 상기 제2레벨 인터벌 내인 경우, 상기 데이터 스트림 내에 기호들의 상기 제3집합을 삽입하도록 구성되는 삽입기(inserter, 40);를 포함하며,

상기 복수의 변환 계수들은 픽처의 블록을 상이한 공간적 주파수들로 분해하는 변환을 표현하는 변환 계수 블록에 포함되는 것을 특징으로 하는,

변환 계수 레벨들을 갖는 복수의 변환 계수들을 데이터 스트림(32)으로 코딩하는 장치.

청구항 18

제17항에 있어서,

상기 장치는 상기 함수(52)가,

$g(f(\mathbf{x}))$ 및 $g(x)=\sum_{i=1}^{d_f}\delta(x,\eta_i)$ 및 $f(x)=\sum_{i=1}^{d} w_i \cdot h \cdot \delta(x_i,t)$ 이며

이와 함께

$$\delta(x,t)=\begin{cases}1 & |x|\geq t\\ 0 & |x|<t\end{cases}$$ 및 $$\delta(x,n)=\begin{cases}1 & x>n\\ 0 & x\leq n\end{cases}$$

이도록 구성되며,

여기서 t, w_i 및 {n_1, ..., n_{d_f}} = n 는 함수 파라미터를 형성하고,

x_i 에서 $i\in\{1...d\}$인 $x=\{x_1,...,x_d\}$ 는 이전에 코딩된 변환 계수 i를 나타내며,

w_i 는 각각이 일과 동일하거나 일과 동일하지 않을 수 있는 가중 값들(weighting values)이며, h는 상수 또는 x_i의 함수인 것을 특징으로 하는 장치.

청구항 19

삭제

청구항 20

제18항에 있어서,

상기 기호화 파라미터 결정기(38)는 기호화 파라미터가 상기 함수를 통해 상기 이전에 코딩된 변환 계수들에 대해 의존적으로 구성되며,

x_i가 이전에 코딩된 변환 계수 i의 변환 계수 레벨과 동일하며, 제1 또는 제2레벨 인터벌 내에 있는 이전에 코딩된 변환 계수 i의 변환 계수 레벨로부터 독립적인 것을 특징으로 하는 장치.

청구항 21

제18항에 있어서,

상기 장치는 $n_1<...<n_{d_f}$인 것을 특징으로 하는 장치.

청구항 22

제18항에 있어서,

상기 장치는 h 가 $|x_i|-t$ 인 것을 특징으로 하는 장치.

청구항 23

제17항에 있어서,

상기 장치는 상기 현재 변환 계수에 관련된 상기 변환 계수들의 상대적 공간적인 배치에 의존하여 상기 이전에 코딩된 변환 계수들을 공간적으로 결정하도록 구성되는 것을 특징으로 하는 장치.

청구항 24

제17항에 있어서,

상기 장치는 상기 데이터 스트림(32)으로부터 미리 결정된 스캔 순서(14)를 따라 변환 계수 블록의 변환 계수들

중 최종 비-제로(non-zero) 변환 계수의 위치에 대한 정보를 결정하도록, 그리고 상기 위치상의 정보를 상기 데이터 스트림으로 삽입하도록 구성되며, 상기 복수의 변환 계수들은 상기 최종 비-제로 변환 계수에서부터 상기 미리 결정된 스캔 순서의 시작에 이르기까지의 변환 계수들을 포함하는 것을 특징으로 하는 장치.

청구항 25

제24항에 있어서,

상기 컨텍스트 적응 엔트로피 인코더는 상기 최종 비-제로 변환 계수 외의 하나 이상의 기호들의 상기 제1집합을 엔트로피 인코딩하는데 이용되는 컨텍스트들과 구분된, 상기 최종 비-제로 변환 계수에 대한 하나 이상의 기호들의 상기 제1집합을 엔트로피 인코딩하기 위한 컨텍스트들의 개별 집합을 이용하도록 구성되는 것을 특징으로 하는 장치.

청구항 26

제24항에 있어서,

상기 컨텍스트 적응 엔트로피 인코더는 상기 최종 비-제로 변환 계수로부터 미리 결정된 스캔 순서의 시작까지 이끄는 반대 스캔 순서로 상기 복수의 변환 계수들을 검색하는 것을 특징으로 하는 장치.

청구항 27

제17항에 있어서,

깊이 맵(depth map)을 상기 데이터 스트림으로 인코딩하도록 구성되는 것을 특징으로 하는 장치.

청구항 28

데이터 스트림(32)으로부터 변환 계수 레벨을 갖는 복수의 변환 계수들(12)을 디코딩하는 방법에 있어서,

현재 변환 계수(x)에 대해, 데이터 스트림(32)으로부터 하나 이상의 기호들의 제1집합(44)을 엔트로피 디코딩하는 단계;

제1기호화 설계에 따라 제1레벨 인터벌(16) 내에서 변환 계수 레벨 상에 하나 이상의 기호들의 제1집합(44)을 역기호화 맵핑(desymbolization mapping)하는 단계;

만약, 하나 이상의 기호들의 제1집합이 제1기호화 설계에 따라 맵핑되는 변환 계수 레벨이 제1레벨 인터벌(16)의 최대 레벨인 경우, 상기 데이터 스트림(32)으로부터 기호들의 제2집합을 추출하는 단계를 포함하며,

상기 역기호화 맵핑하는 단계는 기호화 파라미터(46)에 따라 파라미터화 가능한 제2기호화 설계에 따라 제2레벨 인터벌(18) 내의 위치 상에 기호들의 제2집합(42)을 맵핑하는 것을 포함하며,

제1설정으로 설정되는 함수 파라미터와 함께, 함수 파라미터를 통해 파라미터화 가능한 함수(52)를 통해, 상기 엔트로피 디코딩은 이전에 디코딩된 변환 계수들 상에, 의존하는 컨텍스트를 이용하여 상기 데이터 스트림(32)으로부터 하나 이상의 기호들의 제1집합(44)의 적어도 하나의 미리 결정된 기호를 엔트로피 디코딩하는 것을 포함하며,

하나 이상의 기호들의 제1집합(44)이 제1기호화 설계에 따라 맵핑되는 변환 계수 레벨이 제1레벨 인터벌(16)의 최대 레벨인 경우, 제2설정으로 설정되는 함수 파라미터를 갖는 함수(52)를 통해, 이전에 디코딩된 변환 계수들에 의존하여 상기 기호화 파라미터(46)를 결정하는 것을 더 포함하며,

상기 복수의 변환 계수들은 픽쳐의 블록을 상이한 공간적 주파수들로 분해하는 변환을 표현하는 변환 계수 블록에 포함되는 것을 특징으로 하는,

데이터 스트림(32)으로부터 변환 계수 레벨을 갖는 복수의 변환 계수들(12)을 디코딩하는 방법.

청구항 29

변환 계수 레벨들을 갖는 복수의 변환 계수들을 데이터 스트림(32)으로 코딩하는 방법에 있어서,

현재 변환 계수의 변환 계수 레벨이 제1레벨 인터벌(16) 내에 있는 경우, 제1기호화 설계에 따라 하나 이상의 기호들의 제1집합 상에,

현재 변환 계수의 변환 계수 레벨이 제2레벨 인터벌(18) 내에 있는 경우, 기호화 파라미터(46)에 따라 파라미터화 가능한 제2기호화 설계에 따라, 제2레벨 인터벌(18) 내의 현재 변환 계수의 변환 계수 레벨의 위치에 의존하는 기호들의 제3집합, 및 제1레벨 인터벌(16)의 최대 레벨이 제1기호화 설계에 따라 맵핑되는 기호들의 제2집합의 조합 상에,

현재 변환 계수를 기호화 맵핑하는 단계;

현재 변환 계수들의 변환 계수 레벨이 상기 제1레벨 인터벌 내에 있는 경우, 하나 이상의 기호들의 제1집합을 데이터 스트림으로 엔트로피 인코딩하는 것을 포함하고, 현재 변환 계수의 변환 계수 레벨이 제2레벨 인터벌 내에 있는 경우, 하나 이상의 기호들의 제2집합을 데이터 스트림으로 엔트로피 인코딩하는 것을 포함하는 컨텍스트 적응 엔트로피 인코딩하는 단계;

여기서 상기 컨텍스트 적응 엔트로피 인코딩은, 하나 이상의 기호들의 제2집합의 적어도 하나의 미리 결정된 기호들을 데이터 스트림으로 엔트로피 인코딩하는 데 있어, 제1설정으로 설정되는 함수 파라미터와 함께, 함수 파라미터를 통해 파라미터화 가능한 함수를 통해, 이전에 코딩된 변환 계수들에 의존하는 컨텍스트를 이용하며;

현재 변환 계수의 변환 계수 레벨이 제2레벨 인터벌 내에 있는 경우, 제2설정으로 설정되는 함수 파라미터를 갖는 함수를 통해, 이전에 코딩된 변환 계수들에 의존하는 기호들의 제3집합 상에 맵핑하기 위한 기호화 파라미터(46)를 결정하는 단계;

만약 현재 변환 계수들의 변환 계수 레벨이 제2레벨 인터벌 내에 있는 경우, 데이터 스트림으로 기호들의 제3집합을 삽입하는 단계;를 포함하며,

상기 복수의 변환 계수들은 픽처의 블록을 상이한 공간적 주파수들로 분해하는 변환을 표현하는 변환 계수 블록에 포함되는 것을 특징으로 하는,

변환 계수 레벨들을 갖는 복수의 변환 계수들을 데이터 스트림(32)으로 코딩하는 방법.

청구항 30

제29항에 따른 방법에 의해 발생되는 데이터 스트림을 저장한 컴퓨터 판독 가능한 저장 매체.

청구항 31

제30항에 있어서,

상기 데이터 스트림이 깊이 맵(depth map)을 포함하는 것을 특징으로 하는 컴퓨터 판독 가능한 저장 매체.

청구항 32

삭제

청구항 33

삭제

청구항 34

삭제

청구항 35

삭제

청구항 36

삭제

청구항 37

삭제

청구항 38

삭제

청구항 39

삭제

청구항 40

삭제

청구항 41

삭제

청구항 42

삭제

청구항 43

삭제

청구항 44

삭제

청구항 45

삭제

청구항 46

삭제

청구항 47

삭제

청구항 48

삭제

청구항 49

삭제

청구항 50

삭제

청구항 51

삭제

청구항 52

삭제

청구항 53

삭제

청구항 54

삭제

청구항 55

삭제

청구항 56

삭제

청구항 57

삭제

발명의 설명

기 술 분 야

[0001] 본 발명은 픽쳐(화면, 그림, picture)의 변환 계수 블록의 변환 계수들처럼 변환 계수 코딩과 관련된다.

배 경 기 술

[0002] 블록-기반 이미지 및/또는 비디오 코덱들, 화면 또는 프레임은 블록들 단위로 코딩된다. 동일한 것들 중에서, 변환-기반 코덱은 화면 또는 프레임의 블록들을 변환 대상으로 하며 이는 변환 계수 블록들을 얻기 위함이다. 예를 들어, 그림 또는 프레임은 블록들의 유닛으로 변환 코딩된 예측 잔류와 함께 예측적으로 코딩될 수 있고 이후 코딩은 엔트로피 코딩을 이용하여 이러한 변환 블록들의 변환 계수들의 변환 계수 레벨들을 도출한다.

[0004] 엔트로피 코딩의 효율성을 증가시키기 위해, 컨텍스트들은 코딩된 변환 계수 레벨들의 기호들의 개연성을 정확히 추정하기 위해 이용된다. 그러나, 최근 몇년간, 화면 및/또는 이미지 코덱들에 대한 요구는 증가하였다. 루마(luma) 및 채도 구성요소들에 더하여, 코덱들은 때때로 깊이 맵들(depth maps), 투과도 값들 등등을 운반해야 한다. 게다가, 변환 블록 크기들은 증가적으로 큰 인터벌 내에서 다양하다. 이러한 다양성 때문에, 코덱들은 이미 코딩된 변환 계수들로부터 컨텍스트를 결정하는 데 있어 상이한 함수들을 갖는 상이한 컨텍스트들의 증가하는 숫자를 갖는다.

[0006] 더 적절한 복잡성에서 고압축률을 달성할 상이한 가능성은, 가능한 정확히 계수들의 통계에 기호화 설계를 적응시킨다. 그러나, 실제 통계에 가까운 적응을 수행하기 위해, 다양한 요소들을 반드시 고려해야 하고 그래서 상이한 기호화 설계의 엄청난 양을 필요로 하게 된다.

[0008] 따라서, 높은 코딩 효율을 달성할 가능성을 유지하는 동안에도 불구하고 변환 계수 코딩의 복잡성을 낮게 유지할 필요성이 있다.

발명의 내용

해결하려는 과제

[0009] 그러한 변환 계수 코딩 설계를 제공하는 것이 본 발명의 목적이다.

과제의 해결 수단

[0010] 이 목적은 독립항의 주제에 의해 달성된다.

발명의 효과

[0011] 더 자세히 아래에 설명될 것처럼, 아래에서 더 설명될 변환 계수들을 코딩하는 실시예들은 공통 함수가 동기화 파라미터 결정 및 컨텍스트 적응을 달성하기 위해 이용된다는 점에서 유리하다. 정확한 컨텍스트를 선택하는 것은, 아래에서 설명되는 것처럼, 높은 코딩 효율 또는 압축률을 달성하기 위해 중요하고, 동일한 것들이 기호화 파라미터에 관해 적용된다. 아래에서 설명되는 실시예들은 이전에 코딩된/디코딩된 계수에 대한 의존도를 예시화하기 위한 오버헤드(overhead)를 낮게 유지하는 목적을 달성하는 것을 가능하게한다. 특히, 본 출원의 발명자들은 한편으로는 이전에 코딩된/디코딩된 계수들 상의 효율적인 의존도를 실현하고 다른 한편으로 개별 컨텍스트 의존들을 예시화하는 독점적인 로직(logic)의 숫자를 감소시키는 것 사이의 좋은 타협을 찾았다.

도면의 간단한 설명

[0012] 본 발명의 자세하고 유리한 점들은 종속항의 대상이다. 게다가, 본 발명의 바람직한 실시예들은 다음 도면들과 관련하여 아래에서 설명된다:

도 1은 본 발명의 실시예에 따라 코딩될 변환 계수를 포함하는 변환 계수 블록의 개략적 다이어 그램을 보여주며 기호화 파라미터 결정 및 컨텍스트 선택에 대해 파라미터화 가능한 함수의 동시-사용을 나타낸다.

도 2는 두개의 레벨 인터벌들 내에서 두개의 상이한 설계들을 이용하여 변환 계수 레벨들에 대한 기호화 개념의 개략적 다이어그램을 보여준다.

도 3은 두개의 상이한 컨텍스트들에 대해 가능한 변환 계수 레벨들에 대해 정의되는 두개의 외관 개연성 커브들(appearance probability curves)의 개략적 그래프를 보여준다.

도 4는 실시예에 따라 복수의 변환 계수들을 코딩하는 장치의 블록 다이어그램을 보여준다.

도 5a 및 도 5b는 상이한 실시예에 따라 도출하는 데이터 스트림에 대한 구조의 개략적 다이어그램을 보여준다.

도 6은 실시예에 따라 픽쳐 인코더의 블록 다이어그램을 보여준다.

도 7은 실시예에 따라 복수의 변환 계수들을 디코딩하는 장치의 블록 다이어그램을 보여준다.

도 8은 실시예에 따라 픽쳐 디코더의 블록 다이어그램을 보여준다.

도 9는 실시예에 스캔 및 템플릿을 보여주기 위해 변환 계수 블록의 개략적 다이어그램을 보여준다.

도 10은 추가 실시예에 따라 복수의 변환 계수들을 디코딩하는 장치의 블록 다이어그램을 보여준다.

도 11a 및 도 11b는 및 전체 인터벌 범위의 부분 인터벌들 내에서 두 세개의 상이한 설계들을 결합하는 변환 계수 레벨들에 대한 기호화 개념들의 개략적 다이어그램을 보여준다.

도 12는 추가 실시예에 따라 복수의 변환 계수들을 코딩하는 장치의 블록 다이어그램을 보여준다.

도 13은 추가 실시예에 따른, 변환 계수 블록이 기호화 파라미터 결정 및 컨텍스트 선택에 대한 파라미터화 가능한 함수를 설계하기 위해 또다른 실시예를 도시하기 위해 세분되는 서브-블록들 중 정의되는 서브-블록 순서를 따르는 변환 계수 블록들 중 스캔 순서를 도시하기 위해 변환 계수 블록의 개략적 다이어그램을 보여준다.

발명을 실시하기 위한 구체적인 내용

[0013] 본 발명의 관점에 따라, 데이터 스트림(32)으로부터 변환 계수 레벨을 갖는 다수의 변환 계수들(12)을 디코딩하는 장치는, 데이터 스트림(32)으로부터 하나 이상의 기호들의 제1집합(44)을, 현재 변환 계수에 대해, 엔트로피 디코딩하도록 구성되는 컨텍스트 적응 엔트로피 디코더(80), 제1기호화 설계에 따라 제1레벨 인터벌(16) 내에서 변환 계수 레벨 상에 하나 이상의 기호들의 상기 제1집합(44)을 맵핑하도록 구성되는 역기호화기(82), 하나 이상의 기호들의 제1집합이 제1기호화 설계에 따라 맵핑되는 변환 계수 레벨이 상기 제1레벨 인터벌의 최대 레벨

인 경우, 상기 데이터 스트림(32)으로부터 기호들의 제2집합(42)을 추출하도록 구성되는 추출기(84)를 포함하며, 여기서 상기 역기호화기(82)는 기호화 파라미터에 따라 파라미터화 될 수 있는 제2기호화 설계에 따라 제2레벨 인터벌(18) 내에서의 위치에서 기호들의 제2집합을 맵핑하도록 구성되며, 게다가 본 장치는 상기 컨텍스트 적응 엔트로피 디코더(80)는, 상기 데이터 스트림(32)으로부터 하나 이상의 기호들의 상기 제1집합(44)의 적어도 하나의 미리 결정된 기호를 엔트로피 디코딩하는데 있어, 함수 (52) 파라미터를 통해 파라미터화 될 수 있는 함수를 통해, 제1설정으로 상기 함수 파라미터를 설정하는 것과 함께, 이전에 디코딩된 변환 계수들에 의존하는 컨텍스트를 이용하도록 구성된다. 게다가 본 장치는 하나 이상의 기호들의 상기 제1집합(44)이 상기 제1기호화 설계에 따라 맵핑되는 변환 계수 레벨이 상기 제1레벨 인터벌(16)의 최대 레벨인 경우, 상기 함수 파라미터가 제2설정으로 설정되는 것과 함께 상기 함수(52)을 통해, 상기 이전에 디코딩된 변환 계수들에 의존하는 상기 기호화 파라미터(46)를 결정하도록 구성되는 기호화 파라미터 결정기(86)를 더 포함한다.

[0015] 삽입기는, 현재 변환 계수의 변환 계수 레벨이 제2레벨 인터벌 내에 있는 경우, 기호들의 제3집합을 데이터 스트림으로 삽입하도록 구성된다.

[0017] 본 발명의 또다른 관점에 따르면, 데이터 스트림(32)으로, 각각이 변환 계수 레벨을 갖는, 상이한 변환 블록들의 복수의 변환 계수들(12)를 코딩하기 위한 장치는 기호들의 집합 상에, 기호화 파라미터에 따라 파라미터화 가능한 기호화 설계에 따라 현재 변환 계수에 대한 변환 계수 레벨을 맵핑하도록 구성되는 기호화기, 상기 데이터 스트림에 상기 현재 변환 계수에 대한 기호들의 집합을 삽입하도록 구성되는 삽입기 및 함수 파라미터(46)을 통해 파라미터화 가능한 함수(52)를 통해, 이미 처리된 변환 계수들 상에 의존하여 상기 현재 변환 계수에 대한 기호화 파라미터(46)을 결정하도록 구성되는 기호화 파라미터 결정기를 포함하며, 상기 삽입기(inserter), 기호화기(symbolizer) 및 기호화 파라미터 결정기(symbolization parameter determinator)는 상이한 변환 블록들의 변환 계수들을 순차적으로 처리하도록 구성되며, 현재 변환 계수는 변환 블록 내에 위치되는 것을 특징으로 하는, 데이터 스트림(32)으로, 각각이 변환 계수 레벨을 갖는다.

[0019] 이전에 코딩된/디코딩된 변환 계수들 상에 기호화 파라미터의 의존도 및 컨텍스트의 의존도에 대해 동일한 함수를 이용하는 것이 여기에 이용된 아이디어이다. 동일 함수를 이용하는 것은 - 다양한 함수 파라미터를 변화시키는 것과 함께 - 변환 블록들에 공간적으로 배치되는 변환 계수들의 경우에 변환 블록들의 상이한 변환 블록 크기들 및/또는 주파수 부분들에 관해 이용될 수도 있다. 현재 변환 계수가 변환 블록 내에 위치하며, 현재 변환 계수 블록의 상이한 크기들, 현재 변환 계수의 변환 블록의 상이한 정보 구성요소 타입들 및/또는 상이한 주파수 부분들에 대한 이전에 코딩된/디코딩된 변환 계수들 상의 기호화 파라미터의 의존도에 대해 동일한 함수를 이용하는 것이 이 아이디어의 추가 변형례이다.

[0021] 아래 설명과 관련하여, 동일 도면 부호는 이러한 도면들 중 하나 이상에서 나타나는 구성요소들에 대해 이용된다는 것이 주목된다. 따라서, 하나의 도면에 관한 요소의 설명은 이 요소가 나타나는 또다른 도면의 설명에서 동일하게 적용될 것이다.

[0023] 게다가, 아래에서 제시되는 설명은 그림의 변환 블록처럼 변환 블록을 형성하기 위해 이차원적으로 배치되는 코딩될 변환 계수를 예비적으로 가정한다. 그러나, 본 응용은 이미지 및/또는 비디오 코딩에 제한되지 않는다. 오히려, 코딩될 변환 계수들은, 대안적으로, 예를 들어, 오디오 코딩 또는 유사한 것들에 있어, 이용되는 일차원 변환의 변환 계수들일 수 있다.

[0025] 상기 문제들을 설명하기 위해 아래에서 더 설명되는 실시예들 및 이러한 문제들을 극복하기 위해 아래에서 설명되기 위한 방법들은, 도 1 내지 3이 참고로 제공되며, 이는 이후 설명되는 실시예들에 의해 향상된 엔트로피 코딩의 일반적인 방법 및 변환 블록의 변환 계수들의 예를 보여준다.

[0027] 도 1은 변환 계수들 (12)의 블록(10)을 예시적으로 보여준다. 본 실시예에서, 변환 계수들은 이차원으로 배치된다. 특히, 비록 또다른 이차원 배치가 가능하지만, 동일한 것들이 행과 열로 규칙적으로 배치되는 것이 예시적으로 보여진다. 변환 계수들(12) 또는 변환 블록(10)으로 이끄는 상기 변환은 예를 들어, 공간적으로 배치된 값들의 상이한 공간적 주파수의 구성요소들로의 공간적으로 배치된 값들의 몇몇 다른 블록들 또는 픽쳐의 (변환) 블록을 분해하는 DCT 또는 몇몇 다른 변환일 수 있다. 도 1의 예에서, 변환 계수들 (12)은 열 i 및 행 j로 이차원으로 배치되며 이는 서로 직각 방향처럼 상이한 고간적 방향들을 따라 측정되는 주파수 $f_x(i)$, $f_y(j)$ 의 주파수 쌍들 ($f_x(i)$, $f_y(j)$)에 대응하기 위함이며, 여기서 $f_{x/y}(i) < f_{x/y}(i + 1)$ 및 (i,j) 는 변환 블록 (10)에서 각 계수의 위치이다.

[0029] 더 낮은 주파수들에 대응하는 변환 계수들(12)은 종종 더 높은 주파수들에 대응하는 변환 계수들과 비교하여 더

높은 변환 계수 레벨들을 갖는다. 따라서, 종종 변환 블록(10)의 가장 높은 주파수 구성요소 근처의 변환 계수들 중 많은 것들이 0으로 양자화되고 코딩되어야 할 필요가 없을 수 있다.

[0031] 오히려, 스캔 순서(14)는 이차원으로 배치된 변환 계수들(12) (i, j)를 즉 (i,j)->k인 순서로 계수들의 시퀀스에 일차원적으로 배치하는 변환 계수들 (12) 중에서 정의될 수 있고, 변환 계수 레벨들은 이러한 순서로 즉, 계수 k의 계수 레벨이 계수 k+1 의 계수 레벨보다 큰 순서를 따라 점차 감소하는 경향을 가질 확률이 크다.

[0033] 예를 들어, 지그재그(zigzag) 또는 래스터(raster) 스캔은 변환 계수들(12) 중에서 정의될 수 있다. 스캔에 따라, 블록(10)은, 예를 들어, DC 구성요소 변환 계수 (상측 왼쪽 계수) 에서 최고 주파수 변환 계수 (하측 오른쪽 계수) 또는 그 역으로 대각선으로 스캔될 수 있다. 대안적으로, 방금 언급된 극단 구성요소 변환 계수들 (extreme component transform coefficients) 사이 변환 계수들의 줄-방향 또는 열-방향 스캔이 이용될 수 있다.

[0035] 아래에서 더 설명될 것처럼, 변환 블록을 코딩하는데 있어 스캔 순서(14)로 최종 비-제로 변환 계수 L 의 위치는 데이터 스트림으로 먼저 코딩될 수 있고, 이는 DC 변환 계수로부터 스캔 경로 (14)를 따라 최종 비-제로 변환 계수 L로 - 선택적으로 그러한 방향으로 또는 반대 방향으로 - 변환 계수를 단지 코딩하는 것과 함께이다.

[0037] 변환 계수(12)는 부호를 지닐 수 있는 또는 부호를 지니지 않을 수 있는 변환 계수 레벨들을 가진다. 예를 들어, 변환 계수들(12)은 각각이 각 변환 계수 레벨과 관련된 가능한 양자화 값들 상의 이후 양자화와 함께 앞서 언급된 변환에 의해 얻어졌을 수 있다. 변환 계수들을 양자화하기 위해 이용된 양자화 함수는, 즉 변환 계수 레벨들 상에 변환 계수를 맵핑하는 것은, 선형 또는 비-선형일 수 있다. 다른 말로, 각 변환 계수 (12)는 가능한 레벨들의 인터벌로부터의 변환 계수 레벨을 갖는다. 도 2는, 예를 들어, 변환 계수 레벨들 x 가 레벨들 범위 $[0, 2^{N-1}]$ 내에서 정의되는 예를 보여준다. 대안적 예에 따라, 인터벌 범위의 상측 경계가 없을 수 있다. 게다가, 도 2는 오직 동일한 것이 부호화될 수 있음에도 불구하고 양(positive) 변환 계수 레벨들만이 도시된다. 변환 계수(12)의 부호 및 그들의 코딩과 관련하여, 이러한 부호들을 코딩하기 위해 아래에서 설명되는 실시예들 모두과 관련하여 상이한 가능성들이 존재한다는 것이 주목되어야 하며, 이러한 가능성들 모두가 이러한 실시예들의 범위 내여야 한다. 도 2와 관련하여, 이는 변환 계수 레벨들의 범위 인터벌의 낮은 쪽 경계도 없을 수 있다는 것을 의미한다.

[0039] 어떠한 경우에도, 변환 계수들(12)의 변환 계수 레벨들을 코딩하기 위해, 상이한 기호화 설계(symbolization schemes)들이 범위 인터벌(20)의 상이한 위치들 또는 인터벌들(16, 18)을 커버하기 위해 이용된다. 더 정확하게, 제1레벨 인터벌(16) 내의 변환 계수 레벨들은, 제1레벨 인터벌(16)의 최대 레벨과 동일한 것을 제외하고, 제1기호화 설계에 따라 하나 이상의 기호들의 집합 상에 단순히 기호화될 수 있다. 제2레벨 인터벌(18) 내에 있는 변환 계수 레벨들은, 그러나, 제1 및 제2 기호화 설계들의 기호 집합들의 조합(combination) 상에 맵핑된다. 이후에 설명되는 것처럼, 제3 및 추가 인터벌들은 그에 맞춰 제2인터벌을 따를 수 있다.

[0041] 도 2에서 보여지는 것처럼, 제2레벨 인터벌(18)은 제1레벨 인터벌(16) 위에 위치하지만, 도 2의 예에서 2인, 제1레벨 인터벌(16)의 최대 레벨에서 후자(the latter)와 함께 중첩된다. 제2레벨 인터벌(18) 내에 위치한 변환 계수 레벨들에 대하여, 각 레벨은 제1기호화 설계에 따라 제1레벨 인터벌의 최대 레벨에 대응하는 제1기호 집합, 및 제2기호화 설계에 따라 제2레벨 인터벌 내의 변환 계수 레벨의 위치에 의존하는 제2기호 집합의 조합 상에 맵핑된다.

[0043] 다른 말로, 제1기호화 설계(16)는 제1기호화 시퀀스들 상에 제1레벨 인터벌(16)에 의해 커버되는(걸치는) 레벨들을 맵핑한다. 제1기호화 설계의 기호 시퀀스들의 집합 내의 기호 시퀀스들의 길이는 이진 알파벳의 경우에 그리고 0 및 1처럼 두개의 변환 계수 레벨들만을 아우르는 제1레벨 인터벌(16)의 경우에 단지 하나의 이진 기호일 수 있다. 본 응용의 실시예에 따라, 제1기호화 설계는 인터벌(16)에서 레벨들의 절단 단항 이진일 수 있다. 이진 알파벳의 경우에, 기호들은 호출된 빈들(bins)일 수 있다.

[0045] 아래에서 더 자세히 설명될 것처럼, 제2기호화 설계는 변화하는 길이의 제2기호 시퀀스들의 집합 상에 제2레벨 인터벌(18) 내의 레벨들을 맵핑하며 여기서 제2기호화 설계는 기호화 파라미터에 따라 파라미터화 가능하다 (parameterizable). 제2기호화 설계는 인터벌(18) 내의 레벨들, 즉 x - 제1인터벌의 최대 레벨을, 라이스 파라미터(Rice parameter)를 갖는 라이스 코드(Rice code) 상에 맵핑할 수 있다.

[0047] 특히, 제2기호화 설계(18)는 기호화 파라미터가 제2설계의 기호 시퀀스들의 길이가 제2레벨 인터벌(18)의 낮은 쪽 경계에서 그것의 높은 쪽 경계로 증가하는 비율로 변화하도록 구성될 수 있다. 명백하게, 기호 시퀀스들의 증가된 길이는 변환 계수들이 코딩될 데이터 스트림 내에서 더 많은 데이터 레이트(data rate)를 소모한다. 일

반적으로, 특정 레벨이 맵핑되는 기호 시퀀스의 길이가 실제 개연성과 관련되는 경우 거기서 현재 코딩될 변환 계수 레벨은 각 레벨로 가정된다. 자연스럽게, 나중의 언급은 제1레벨 인터벌(16) 내의 또는 일반적으로 제1기호화 설계에 대해 제2레벨 인터벌(18) 외곽의 레벨들에 대해서도 유효하다. 특히, 도 3에서 보여지는 것처럼, 변환 계수들은 일반적으로 특정 변환 계수 레벨들의 발생의 개연성 또는 특정 통계들을 보여준다. 도 3은 각 가능한 변환 계수 레벨 x 각 변환 계수 레벨이 논의되고 있는 변환 계수에 의해 실제로 가정되는 개연성과 관련된 그래프를 보여준다. 더 자세히, 도 3은 즉, 상이한 컨텍스트들의 두 계수들에 대한, 두개의 그러한 관련들 또는 개연성 커브들을 보여준다. 그것은, 도 3은 인접 변환 계수들의 변환 계수 값들에 의해 결정되는 것처럼 그들의 컨텍스트들에 따라 구분되는 변환 계수들을 가정한다. 컨텍스트에 의존하여, 도 3은 각 변환 계수 레벨과 개연성 값을 관련시키는 개연성 커브(curve)가 논의 중인 변환 계수의 컨텍스트에 의존할 수 있다는 것을 보여준다.

[0049] 아래에서 설명되는 실시예에 따라, 제1기호화 설계(16)의 기호 시퀀스들의 기호들은 컨텍스트 적응 방식으로 엔트로피 코딩된다. 컨텍스트는 기호들과 관련되며, 선택된 컨텍스트와 관련된 알파벳 개연성 분포는 각 기호들을 엔트로피 코딩하는데 이용된다. 제2기호화 설계의 기호 시퀀스들의 기호들은 데이터 스트림으로 바로 또는 고정된 알파벳 개연성 분포를 이용하여 삽입되며 알파벳의 모든 멤버들에 따른 동일한 개연성 분포는 동등한 개연성을 가진다.

[0051] 제1기호화 설계의 기호들을 엔트로피 코딩하는데 이용되는 컨텍스트들은 실제 알파벳 통계들에 추정된 알파벳 개연성 분포의 좋은 적응을 허용하기 위해 적절히 선택되어야 한다. 그것은, 엔트로피 코딩 설계는 이 컨텍스트를 갖는 기호들이 인코딩/디코딩될 때마다 컨텍스트의 알파벳 개연성 분포의 현재 추정을 업데이트하도록 구성될 수 있다는 것이고, 그래서 실제 알파벳 통계들을 추정한다. 상기 추정은 만약 컨텍스트들이 적절히 선택되고 충분히 세밀하다면 더 빠르지만, 그것은 특정 컨텍스트들과 기호들의 너무 드문 관련을 피하기 위해 많은 상이한 컨텍스트들과는 아니다.

[0053] 비슷하게, 계수에 대한 기호화 파라미터는 가능한 근접하여 실제 알파벳 통계를 추정하기 위해 이전에 코딩된/디코딩된 계수들에 의존하여 선택되어야 한다. 너무 미세한 다양화는 여기서 중요한 이슈가 아니며, 이는 기호화 파라미터가 이전에 코딩된/디코딩된 계수들로부터 직접 결정되기 때문이며, 다만 상기 결정은 이전에 코딩된/디코딩된 계수들 상에 제2인터벌(18) 내에서 개연성 커브의 의존도의 상관에 밀접하게 관련되어야 한다.

[0055] 더 자세히 아래에 설명될 것처럼, 아래에서 더 설명될 변환 계수들을 코딩하는 실시예들은 공통 함수가 동기화 파라미터 결정 및 컨텍스트 적응을 달성하기 위해 이용된다는 점에서 유리하다. 정확한 컨텍스트를 선택하는 것은, 아래에서 설명되는 것처럼, 높은 코딩 효율 또는 압축률을 달성하기 위해 중요하고, 동일한 것들이 기호화 파라미터에 관해 적용된다. 아래에서 설명되는 실시예들은 이전에 코딩된/디코딩된 계수에 대한 의존도를 예시화하기 위한 오버헤드(overhead)를 낮게 유지하는 목적을 달성하는 것을 가능하게한다. 특히, 본 출원의 발명자들은 한편으로는 이전에 코딩된/디코딩된 계수들 상의 효율적인 의존도를 실현하고 다른 한편으로 개별 컨텍스트 의존들을 예시화하는 독점적인 로직(logic)의 숫자를 감소시키는 것 사이의 좋은 타협을 찾았다.

[0057] 도 4는 본 발명의 실시예에 따라 변환 계수 레벨들을 갖는 복수의 변환 계수들을 데이터 스트림으로 코딩하는 장치를 보여준다. 다음 설명에서, 기호 알파벳은 이진 알파벳으로 종종 가정되고, 비록 이러한 가정들은, 위에서 설명되는 것처럼, 본 발명에 중요하지는 않으며 따라서, 이러한 모든 설명들은 다른 기호 알파벳들 상의 확장에 대한 설명으로서 해석되어야 한다.

[0059] 도 4의 장치는 입력 (30)에서 데이터 스트림(32)으로 들어가는 복수의 변환 계수들을 코딩하기 위한 것이다. 상기 장치는 기호화기(34), 컨텍스트 적응 엔트로피 인코더(36), 기호화 파라미터 결정기(38) 및 삽입기(40)를 포함한다.

[0061] 기호화기(34)는 입력(30)과 연결된 그것의 입력을 가지며 도 2와 관련하여 위에서 설명된 방식으로 기호들 상에 그것의 입력을 현재 입력하는 현재 변환 계수를 맵핑하도록 구성된다. 그것은, 기호화기(34)는 만약 현재 변환 계수의 변환 게수 레벨 x가 제1레벨 인터벌(16) 내에 있는 경우, 제1기호화 설계에 따라 하나 이상의 기호들의 제1집합 상에, 그리고 만약 현재 변환 계수의 변환 계수 레벨이 제2레벨 인터벌(18) 내에 있는 경우, 제1기호화 설계에 따라 제1레벨 인터벌(16)의 최대 레벨이 맵핑되는 기호들의 제2집합과, 제2기호화 설계에 따라 제2레벨 인터벌(18) 내의 현재 변환 계수의 변환 계수 레벨의 위치에 의존하는 기호들의 제3집합의 조합 상에, 현재 변환 계수를 맵핑하도록 구성된다는 것이다. 다른 말로, 기호화기(34)는 현재 변환 계수의 변환 계수 레벨이 제1레벨 인터벌(16) 내에 그러나 제2레벨 인터벌 밖에 있는 경우, 제1기호화 설계의 제1기호 시퀀스 상에, 현재 변환 계수의 변환 계수 레벨이 제2레벨 인터벌 내에 있는 경우, 제1레벨 인터벌(16)의 최대 레벨에 대한 제1기호화 설계의 기호 시퀀스 및 제2기호화 설계의 기호 시퀀스의 조합 상에, 현재 변환 계수를 맵핑하도록 구성된다.

[0063] 상기 기호화기(34)는 두개의 출력들을 가지며, 즉 제1기호화 설계의 기호 시퀀스들 중 하나, 및 제2기호화 설계의 기호 시퀀스들 중 또다른 것이다. 삽입기(40)는 제2기호화 설계의 기호 시퀀스들(42)을 수신하기 위한 입력을 가지며 컨텍스트 적응 엔트로피 인코더(36)는 제1기호화 설계의 기호 시퀀스들(44)을 수신하기 위한 입력을 갖는다. 이에 더하여, 기호화기(34)는 기호화 파라미터 결정기(38)의 출력으로부터 기호화 파라미터(46)를 수신하기 위한 파라미터 입력을 갖는다.

[0065] 컨텍스트 적응 엔트로피 인코더(36)는 데이터 스트림(32)으로 제1기호화 시퀀스들(44)의 기호를 엔트로피 인코딩하도록 구성된다. 삽입기(40)는 데이터 스트림으로 기호 시퀀스들(42)을 삽입하도록 구성된다.

[0067] 일반적으로 말해, 양쪽 엔트로피 인코더(36) 및 삽입기(40)는 순차적으로 변환 계수들을 스캔한다. 명백하게, 삽입기(40)는 단지 변환 계수에 대해 작동하며, 그것의 변환 계수 레벨은 제2레벨 인터벌(18) 내에 위치한다. 그러나, 아래에서 더 자세히 설명되는 것처럼, 엔트로피 인코더(36) 및 삽입기(40)의 작동 사이에 순서를 정의하기 위한 상이한 개연성이 있다. 제1실시예에 따라, 도 4의 코딩 장치는 삽입기(40)가 데이터 스트림(32)으로 동일 변환 계수에 관련된 제1기호 시퀀스(44)의 엔트로피 인코더의 엔트로피 인코딩 다음에 그리고 데이터 스트림으로 인라인으로(in line) 다음 변환 계수에 관련된 기호 시퀀스(44)를 엔트로피 인코더의 엔트로피 인코딩에 앞서 데이터 스트림(32)으로 변환 계수의 기호 시퀀스(42)를 삽입하도록 하나의 단일 스캔으로 변환 계수들을 스캔하도록 구성된다.

[0069] 대안적 실시예에서, 상기 장치는 두개의 스캔(scan)을 이용하여, 여기서 제1스캔 내에서 컨텍스트 적응 엔트로피 인코더(36)는 삽입기(40)와 각 변환 계수에 대해 데이터 스트림(32)으로 기호 시퀀스들(44)를 순차적으로 인코딩하며 이후 제2레벨 인터벌(18) 내에 있는 변환 계수 레벨의 변환 계수들에 대해 기호 시퀀스들(42)을 삽입한다. 그것에 따른 더 복잡한 설계들이 있을 수 있으며, 예를 들어, 컨텍스트 적응 엔트로피 인코더(36)는 제2스캔 등등에서 시퀀스들(44)의 빈(bin) 또는 제2기호가 뒤따르는, 제1스캔에서의 빈 또는 제1기호처럼 데이터 스트림(32)으로 제1기호 시퀀스들(44)의 개별 기호들을 인코딩하기 위해 몇몇 스캔들(scans)을 이용한다.

[0071] 위에서 이미 표시된 것처럼, 컨텍스트 적응 엔트로피 인코더(36)는 컨텍스트 적응 방식으로 데이터 스트림(32)으로 기호 시퀀스들(44)의 적어도 하나의 미리 결정된 기호에서 엔트로피 인코딩하도록 구성된다. 예를 들어, 컨텍스트 적응성은 기호 시퀀스들(44)의 모든 기호들에 대해 이용될 수 있다. 대안적으로, 컨텍스트 적응 엔트로피 인코더(36)는 제1기호화 설계의 기호 시퀀스들 및 제1위치에서, 또는 제1 및 제2, 또는 제1에서 제3위치들 등등의 기호들에만 컨텍스트 적응성을 제한할 수 있다.

[0074] *위에서 설명된 것처럼, 컨텍스트 적응성에 대해, 인코더(36)는 각 컨텍스트에 대해 알파벳 개연성 분포 추정을 업데이트하고 저장하는 것에 의해 컨텍스트들을 관리한다. 매번 특정 컨텍스트의 기호가 인코딩되며, 현재 저장된 알파벳 개연성 분포 추정은 이 기호의 실제 값을 이용하여 업데이트되고 그 컨텍스트의 기호의 실제 알파벳 통계들을 추정한다.

[0076] 이와 유사하게, 기호화 파라미터 결정기(38)는 이전에 코딩된 변환 계수들 상에 의존하는 기호 시퀀스들(42) 및 제2기호화 설계에 대해 기호화 파라미터(46)를 결정하도록 구성된다.

[0078] 더 정확하게, 기호화 파라미터(38)가 제2설정으로 설정되는 함수 파라미터와 함께, 동일 함수를 통해, 이전에 코딩된 변환 계수들 상에, 의존하는 기호화 파라미터(46)를 결정하도록 구성되는 동안, 컨텍스트 적응 엔트로피 인코더(36)는 동일한 것이, 제1설정으로 설정되는 함수 파라미터와 함께, 함수 파라미터를 통해 파라미터화 가능한 함수를 통해, 이전에 코딩된 변환 계수들 상에, 의존하는 컨텍스트를 현재 변환 계수에 대해, 이용하고 선택하도록 구성된다. 상기 설정들은, 그럼에도 불구하고, 기호화 파라미터 결정기(38)과 다를 수 있고, 컨텍스트 적응 엔트로피 인코더(36)는 동일 함수를 이용하며, 로직 오버헤드(logic overhead)는 감소될 수 있다. 단지 함수 파라미터가 한편으로는 엔트로피 인코더(36)의 컨텍스트 선택 및 다른 한편으로 기호화 파라미터 결정기(38)의 기호화 파라미터 결정 사이에서 다를 수 있다.

[0080] 이전에 코딩된 변환 계수들 상의 의존도에 관해서, 이 의존도는 이러한 이전에 코딩된 변환 계수들이 이미 데이터 스트림(32)으로 코딩된 것으로의 확장이 제한된다는 것이 주목되어야 한다. 예를 들어, 이전에 인코딩된 변환 계수가 제2레벨 인터벌(18) 내에 존재하지만, 그것의 기호 시퀀스(42)는 데이터 스트림(32)으로 아직 삽입되지 않았다고 생각해보자. 그러한 경우에, 기호화 파라미터 결정기(38) 및 컨텍스트 적응 엔트로피 인코더(36)는 단지 이전에 코딩된 변환 계수의 제1기호화 시퀀스(44)로부터 동일한 것이 제2레벨 인터벌(18) 내에 위치한다는 것만을 알게된다. 그러한 경우에, 제1레벨 인터벌(16)의 최대 레벨은 이전에 코딩된 변환 계수에 대한 표현으로 기능하게 된다. "이전에 코딩된 변환 계수들"에의 의존도는 "데이터 스트림(32)으로 이전에 인코딩/삽입되는 다

른 변환 계수들 상의 정보"에의 의존도를 포함하기 위해 넓은 의미로 이해되어야 한다. 게다가, 최종 비-제로 계수 L 위치를 "넘어" 존재하는 변환 계수들은 0으로 추론될 수 있다.

[0082] 도 4의 설명을 마무리하기 위해서, 엔트로피 인코더(36) 및 삽입기(40)의 출력은 스위치(50)를 통해 장치의 공통 출력(48)에 연결되는 것으로 보여지며, 이는 한편으로 컨텍스트 적응 엔트로피 인코더(36) 및 기호화 파라미터 결정기(38)의 이전에 삽입/코딩된 정보에 대한 입력들과 다른 한편으로 삽입기(40) 및 엔트로피 인코더(36)의 출력 사이에 존재하는 동일한 연결성을 갖는다. 스위치(50)는 변환 계수들을 코딩하기 위해 하나, 둘 또는 그 이상의 스캔을 이용하는 다양한 개연성과 관련하여 위에서 언급된 순서로 삽입기(40) 및 엔트로피 인코더(36)의 출력 중 각 하나와 출력(48)을 연결한다.

[0084] 더 특정적인 용어들에서 기호화 파라미터 결정기(38) 및 컨텍스트 적응 엔트로피 인코더(36)에 관해 파라미터화 가능한 함수의 일반적인 이용을 설명하기 위해, 도 1이 논의된다. 엔트로피 인코더(36) 및 기호화 파라미터 결정기(38)에 의해 공동으로 이용되는 함수는 도 1에서 (52)로 지칭되고, 즉 $g(f(x))$이다. 상기 함수는, 위에서 설명된대로, 현재 계수에 관련하여 특정 공간적 관계를 가지고 이전에 코딩된 계수들을 아우르도록 정의될 수 있는 이전에 코딩된 변환 계수들의 집합에 적용된다. 이 함수의 특정 실시예들은 아래에서 더 자세히 설명될 것이다. 일반적으로, f는 스칼라(scalar)로 이전에 코딩된 계수 레벨들의 집합을 결합하는 함수이고, g는 스칼라(scalar)가 위치한 인터벌을 확인하는 함수이다. 다른 말로, 함수 $g(f(\boldsymbol{x}))$ 는 이전에 코딩된 변환 계수의 집합 x에 적용된다. 도 1에서, 작은 십자가에 의해 표시되는 변환 계수(12)는, 예를 들어, 현재 변환 계수를 나타내며, 해칭된(빗금쳐진) 변환 계수들(12)은 상기 함수가 기호화 파라미터(46) 및 현재 변환 계수에 대한 컨텍스트를 인덱싱하는 엔트로피 컨텍스트 인덱스(54)를 얻기 위해 적용되는 변환 계수들의 집합 x를 지칭한다. 도 1에서 도시되는 대로, 현재 변환 계수 주변에서 상대적 공간적 배치를 정의하는 로컬 템플릿(local template)은, 모든 이전에 코딩된 변환 계수들로부터 상대적인 이전에 코딩된 변환 계수들의 집합 x를 결정하기 위해 이용될 수 있다. 도 1에서 보여질 수 있는 것처럼, 템플릿(56)은 현재 변환 계수의 오른쪽 및 아래에 바로 인접한 변환 계수를 포함할 수 있다. 이러한 템플릿을 선택하는 것에 의해 스캔(140)의 하나의 대각선에서 변환 계수들의 기호 시퀀스들(42 및 44)는 동일 대각선으로 또다른 변환 계수의 템플릿(56)으로 떨어지는 대각선 상의 변환 계수들이 없기 때문에 병렬로 코딩될 수 있다. 자연스럽게, 유사한 템플릿들이 줄- 및 열- 방향 스캔에 대해 발견될 수 있다.

[0086] 대응 함수 파라미터들 및 공통적으로 이용되는 함수 $g(f(x))$에 대해 더 특정적인 예들을 제공하기 위해, 다음에서 그러한 예들이 각 공식을 이용하여 제공된다. 특히, 도 4의 장치는 한편으로 이전에 코딩된 변환 계수들의 집합 x, 및 다른 한편으로 기호화 파라미터(46) 및 컨텍스트를 인덱싱하는 컨텍스트 인덱스 숫자(54) 사이의 관계를 정의하는 함수가,

[0087] $g(f(\mathbf{x}))$ 및 $g(x)=\sum_{i=1}^{d_f}\delta'(x,n_i)$ 및 $f(\mathbf{x})=\sum_{i=1}^{d}w_i\cdot h\cdot\delta(x_i,t)$

[0088] 와 함께

[0089] $$\delta(x,t)=\begin{cases}1 & |x|\geq t\\ 0 & |x|<t\end{cases}$$ 및 $$\delta(x,n)=\begin{cases}1 & x>n\\ 0 & x\leq n\end{cases}$$

[0090] 이며,

[0091] 여기서 t 및 $\{n_1, \ldots, n_{d_f}\} = n$ 및, 선택적으로 w_i는 함수 파라미터를 형성하고,

[0092] x_i 에서 $i\in\{1\ldots d\}$인 $x=\{x_1,\ldots,x_d\}$ 는 이전 디코딩된 변환 계수 i를 나타내며,

[0093] w_i는 각각이 일과 동일하거나 일과 동일하지 않을 수 있는 가중 값들(weighting values)이며, h는 상수 또는 x_i의 함수일 수 있도록 구성될 수 있다.

[0095] 이는 $g(f(x))$ 가 $[0,d_f]$ 내에 위치한다는 것을 따른다. 만약 $g(f(x))$ 가 적어도 하나의 베이스 컨텍스트 인덱스 오프셋 숫자 ctx_{base} 에 따라 합쳐지는 컨텍스트 인덱스 오프셋 숫자 ctx_{offset} 를 정의하는데 이용된다면, 결과 컨텍스트 인덱스 $ctx= ctx_{base} + ctx_{offset}$ 의 값 범위는 $[ctx_{base} ; ctx_{base} + d_f]$이다. 컨텍스트들의 다른 집합들이 기

호 시퀀스들(44)의 기호들을 엔트로피 코딩하는데 이용된다는 것이 언급될 때마다, ctx_{base} 는 [$ctx_{base,1}$; ctx_{base} + d_f] 이 [$ctx_{base,2}$; ctx_{base} + d_f]를 중첩하지 않도록 다르게 선택된다. 이는, 예를 들어,

[0097] ● 상이한 크기의 변환 블록들에 속하는 변환 계수들

[0098] ● 깊이, 루마(광도, luma), 채도(chroma) 등등처럼 다른 정보 구성요소 타입의 변환 블록들에 속하는 변환 계수들

[0099] ● 동일 변환 블록의 다른 주파수 부분들에 속하는 변환 계수들

[0100] 에 대하여 참(true)이다.

[0102] 이전에 언급되었듯이, 기호화 파라미터는 라이스 파라미터(Rice parameter) k 이다. 그것은, (M 이 인터벌 (16)의 최대 레벨이고 x가 (절대) 변환 계수 레벨인) $X + M = x$ 인 관계에 있는, 인터벌(16) 내에서 (절대적) 레벨들, 즉 X 가 접두사 및 접미사를 갖는 빈 스트링(bin string) 상에 맵핑된다는 것이며, 접두사는 $\lfloor X \cdot 2^{-k} \rfloor$ 의 단항 코드이며, 접미사는 $X \cdot 2^{-k}$ 의 나머지의 이진 코드이다.

[0104] d_f 는 함수 파라미터의 부분을 형성할 수도 있다. d 가 함수 파라미터의 부분을 형성할 수도 있다.

[0106] 컨텍스트 선택 및 기호화 파라미터 결정처럼 함수 파라미터에서의 차이는 각각 t, $\{n_1 < \ldots < n_{d_f}\} = n$, d_f (만약 함수 파라미터의 부분을 형성하는 경우), 또는 d (함수 파라미터의 부분을 형성하는 경우) 단지 하나의 차이를 필요로 한다.

[0108] 위에서 설명된 것처럼, 인덱스 i는 템플릿(56) 내에서 변환 계수들 (12)을 인덱싱(색인화, index) 할 수 있다. x_i 는 각 템플릿 위치가 변환 블록의 바깥에 위치하는 경우에 0(zero)로 설정될 수 있다. 게다가, 컨텍스트 적응 엔트로피 인코더(36)는 함수를 통해 이전에 코딩된 변환 계수들로부터 컨텍스트의 의존도가 동일한 것이 제1 레벨 인터벌(16) 내에 있는 경우 x_i가 이전에 코딩된 변환 계수 i 의 변환 계수 레벨과 동일하도록, 그리고 이전에 코딩된 변환 계수 i 의 변환 계수 레벨이 제2레벨 인터벌(18) 내에 있는 경우 제1레벨 인터벌(16)의 최대 레벨과 동일하도록, 또는 x_i 가, 제1 또는 제2 레벨 인터벌 내에 있는 이전에 코딩된 변환 계수 i의 변환 계수 레벨로부터 독립적인, 이전에 코딩된 변환 계수 i의 변환 계수 레벨과 동일하도록 구성될 수 있다.

[0110] 기호화 파라미터의 결정기에 관해서, 기호화 파라미터의 결정에 있어서, 동일하게 구성될 수 있고, 이는 x_i 가 제1 또는 제2 레벨 인터벌 내에 위치하는 이전에 코딩된 변환 계수의 변환 계수 레벨로부터 독립적인,이전에 코딩된 변환 계수 i 의 변환 계수 레벨과 동일하다는 것이다.

[0112] 상기 장치는 $n_1 < \ldots < n_{d_f}$ 가 어느 경우에나 적용되도록 더 구성될 수 있다는 것이다.

[0114] 상기 장치는 또한 $h = |x_i| - t$ 이도록 구성될 수 있다.

[0116] 추가 실시예에서 상기 장치는 현재 변환 계수에 관련된 변환 계수들의 상대적 공간적 배치에 의존하여, 즉 현재 변환 계수의 위치 주변의 템플릿에 기반하여 이전에 코딩된 변환 계수들을 공간적으로 결정하도록 구성될 수 있다.

[0118] 상기 장치는 미리 결정된 스캔 순서(14)를 통해 변환 계수 블록(10)의 변환 계수들 중에서 최종 비-제로(last non-zero) 변환 계수 L의 위치를 결정하도록 더 구성될 수 있고, 데이터 스트림(32)으로 위치상의 정보를 삽입하도록 구성될 수 있고, 복수의 변환 계수들은, 최종 비-제로 변환 계수 L 에서부터 미리 결정된 스캔 순서의 시작까지, 즉, DC 구성요소 변환 계수까지, 변환 계수들을 포함한다.

[0120] 추가 실시예에서, 기호화기(34)는 최종 변환 계수 L의 기호화에 대한 수정된 제1기호화 설계를 이용하도록 구성될 수 있다. 수정된 제1기호화 설계에 따라, 제로 레벨(zero level)이 최종 변환 계수 L 에 대해 적용하지 않는 것으로 가정되는 동안, 제1레벨 인터벌(16) 내의 비-제로 변환 계수 레벨들만이 맵핑될 수 있다. 예를 들어, 절단 단항 이진화(truncated unary binarization)의 제1빈(first bin)은 계수 L 에 대해 억제될 수 있다.

[0122] 컨텍스트 적응 엔트로피 인코더는, 최종 비-제로 변환 계수 외에 하나 이상의 기호들의 제1집합을 엔트로피 인

코딩하는데 이용된 컨텍스트들과 구분되는, 최종 비-제로 변환 계수에 대한 하나 이상의 기호들의 제1집합을 엔트로피 인코딩하기 위한 컨텍스트들의 개별 집합을 이용하도록 구성될 수 있다.

[0124] 컨텍스트 적응 엔트로피 인코더는 최종 비-제로 변환 계수에서 변환 계수 블록의 DC 변환 계수까지 이끄는 반대 스캔 순서로 복수의 변환 계수들을 검색할 수 있다. 이는 제2기호 시퀀스(42)에 대해서 적용될 수도 적용되지 않을 수도 있다. 상기 장치는 두개의 스캔으로 복수의 변환 계수들을 데이터 스트림(32)으로 코딩하도록 구성될 수 있고, 여기서 상기 컨텍스트 적응 엔트로피 코더(36)는 변환 계수들의 제1스캔에 대응하는 순서로 데이터 스트림(32)으로 변환 계수들에 대한 제1기호 시퀀스들(44)을 엔트로피 인코딩하도록 구성될 수 있으며, 여기서 상기 삽입기(40)는 변환 계수들의 스캔 순서 내에서 제2레벨 인터벌(18) 내의 변환 계수 레벨을 갖는 변환 계수들의 발생에 대응하는 순서로 데이터 스트림(32)으로 제2레벨 인터벌(18) 내의 변환 계수 레벨을 갖는 변환 계수들에 대해 기호 시퀀스들(42)를 순차적으로 삽입하도록 구성된다. 결과 데이터 스트림(32)에 대한 예가 도 5a에서 보여지며 : 그것은, 엔트로피 인코딩된 형태로 (적어도 컨텍스트 적응 엔트로피 인코딩된 형태의 몇몇) 기호 시퀀스들(42)가 뒤따르는 그리고 예를 들어, (동일 가능한 알파벳) 바이패스(bypass) 모드를 이용하거나 직접 삽입되는 기호 시퀀스들(44)가 더 뒤따르는, L의 위치 상의 정보(57)를, 선택적으로, 포함할 수 있다.

[0126] 추가 실시예에서, 상기 장치는 하나의 스캔에서 순차적으로 데이터 스트림(23)으로 복수의 변환 계수들을 코딩하도록 구성될 수 있고, 상기 적응 엔트로피 인코더(36) 및 삽입기(40)는, 하나의 스캔의 스캔 순서로 각 변환 계수에 대해, 동일하게 동일 변환 계수들이 맵핑되는 조합을 형성하는 것을 따라, 데이터 스트림(32)으로 기호 시퀀스(44)의 컨텍스트 적응 엔트로피 코더의 엔트로피 인코딩 바로 다음에 데이터 스트림(32)으로 제2레벨 인터벌(18) 내에 변환 계수 레벨을 갖는 각 변환 계수들의 기호 시퀀스들(42)을 삽입하도록, 구성되며 이는 기호 시퀀스들(42)이 변환 계수들의 기호 시퀀스들(44) 사이의 데이터 스트림(32)으로 배치되도록 하기 위함이다. 상기 결과는 도 5b에서 도시된다.

[0128] 삽입기(40)는 고정된 개연성 분포를 이용하여 엔트로피 인코딩을 이용하거나직접 데이터 스트림으로 기호 시퀀스들(42)를 삽입하도록 구성될 수 있다. 제1기호화 설계는 절단 단항 이진화 설계일 수 있다. 제2기호화 설계는 기호 시퀀스들(42)가 라이스 코드(Rice code)일 수 있다.

[0130] 위에서 이미 알려졌듯이, 도 4의 실시예는 이미지/비디오 코더 내에서 실행될 수 있다. 그러한 이미지/비디오 코더 또는 픽쳐 코더(picture coder)의 예는 도 6에서 보여진다. 픽쳐 인코더는 일반적으로 도면 부호(60)으로 지칭되며 예를 들어, 도 4에서 보여지는 것에 대응하는 장치를 포함한다. 인코더(60)는, 변환 블록(10) 당, 그것의 복수의 변환 계수들을 코딩하기 위해, 픽쳐(그림, 화면, 사진, picture)를 인코딩하는데 있어(64), 장치(62)에 의해 처리되는 변환 계수 블록들(10)로 픽쳐(64)의 블록들(66)을 변환하도록 구성된다. 특히, 장치(62)는 변환 블록별로 변환 블록(10)을 처리한다. 그렇게 함으로써, 장치(62)는 상이한 크기들의 블록들(10)에 대한 함수(52)를 이용할 수 있다. 예를 들어, 계층적 멀티-트리 분할은 상이한 크기들의 블록들(66)로, 픽쳐(64) 또는 그것의 트리-기반 블록(tree-root blocks)을 분해하기 위해 이용될 수 있다. 비록, 함수(52)가 상이한 함수 파라미터들을 이용하는 방식으로 상이한 블록 크기들에 대해 최적화된다 하더라도, 이러한 블록들(66)로의 변환을 적용하는 것으로부터 도출되는 변환 블록들(10)은, 따라서 상이한 크기들이며, 한편으로는 기호화 파라미터에 대한 다른 한편으로는 컨텍스트 인덱스에 대하 그러한 상이한 의존도를 제공하는 전체 오버헤드(overall overhead)가 낮게 유지된다.

[0132] 도 7은 도 4에 관해 위에서 설명된 장치에 맞는 데이터 스트림(32)으로부터 변환 계수 레벨들을 갖는 복수의 변환 계수들을 디코딩하는 장치를 보여준다. 특히, 도 7은 기호화 파라미터 결정기(86) 뿐만 아니라 컨텍스트 적응 엔트로피 디코더(80), 역기호화기(82) 및 추출기(84)를 포함한다. 컨텍스트 적응 엔트로피 디코더(80)는, 현재 변환 계수에 대해, 데이터 스트림(32)으로부터, 즉 기호 시퀀스(44)인, 하나 이상의 기호들의 제1집합을 엔트로피 디코딩하도록 구성된다. 역기호화기(desymbolizer, 82)는 제1기호화 설계에 따라 제1레벨 인터벌(16) 내에서 변환 계수 레벨 상에, 즉 기호 시퀀스(44)인, 하나 이상의 기호들의 제1집합을 맵핑하도록 구성된다. 더 자세히는, 컨텍스트 적응 엔트로피 디코더(80) 및 역기호화기(82)는 상호작용 방식으로 작동한다. 역기호화기(82)는 기호가 데이터 스트림(32)으로부터 디코더(80)에 의해 순차적으로 디코딩된 신호(88)를 통해 컨텍스트 적응 엔트로피 디코더(80)에게 제1기호화 설계의 유효 기호 시퀀스가 종결되었다는 것을 알려준다.

[0134] 추출기(84)는, 하나 이상의 기호들의 제1집합, 즉 기호 시퀀스(44)가 제1기호화 설계에 따라 맵핑된 변환 계수 레벨이 제1레벨 인터벌(16)의 최대 레벨인 경우, 데이터 스트림(32)으로부터 기호 시퀀스(42)인, 기호들의 제2집합을 추출하도록 구성된다. 다시, 역기호화기(82) 및 추출기(84)는 콘서트에서 작동할 수 있다. 그것은, 역기호화기(82)는 추출기(84)가 기호 시퀀스(42)의 추출이 끝날 수 있기 때문에 제2기호화 설계의 유효한 기호 시퀀

스가 종결될 때 신호(90)에 의해 추출기(84)에 알려줄 수 있다는 것이다.

[0136] 역기호화기(82)는, 이미 위에서 언급되었듯이, 기호화 파라미터(46)에 따라 파라미터화 가능한, 제2기호화 설계에 따라 제2레벨 인터벌(18) 내에서의 위치 상에, 기호들의 제2집합, 즉 기호 시퀀스(42)를 맵핑하도록 구성된다.

[0138] 컨텍스트 적응 엔트로피 디코더(80)는, 제1기호 시퀀스(44)의 적어도 하나의 미리 결정된 기호를 엔트로피 디코딩하는데 있어, 함수(52)를 통해, 이전에 디코딩된 변환 계수들 상에 의존하는 컨텍스트를 이용하도록 구성된다. 제1기호화 시퀀스(44)가 제1기호화 설계에 따라 맵핑되는 변환 계수 레벨이 제1레벨 인터벌(16)의 최대 레벨인 경우, 기호화 파라미터 결정기(86)는, 함수(52)를 통해, 이전에 디코딩된 변환 계수들에 의존하여 기호화 파라미터(46)를 결정하도록 구성된다. 이런 이유로, 엔트로피 디코더(80) 및 기호화 파라미터 결정기(86)의 입력들은 스위치(92)를 통해 변환 계수들의 값 x_i를 출력하는 역기호화기(82)의 출력에 연결된다.

[0140] 위에서 설명된 것처럼, 컨텍스트 적응성에 대해, 디코더(80)는 각 컨텍스트에 대해 알파벳 개연성 분포 추정을 저장하고 업데이트하는 것에 의해 컨텍스트를 관리한다. 매번 특정 컨텍스트의 기호가 디코딩되고, 현재 저장된 알파벳 개연성 분포 추정은 그것의 컨텍스트의 기호의 실제 알파벳 통계를 추정함으로써 이 기호의 실제/디코딩된 값을 이용하여 업데이트된다.

[0142] 이와 유사하게, 기호화 파라미터 결정기(86)는 제2기호화 설계에 대해 기호화 파라미터(46) 및 이전에 디코딩된 변환 계수들 상에 의존하여 기호 시퀀스들(42)를 결정하도록 구성된다.

[0144] 일반적으로, 인코딩에 관해 위에서 설명된 모든 가능한 수정 및 추가 자세한 내용들은 도 7의 디코딩 장치 상에도 전환 가능하다.

[0146] 도 8은 도 6에서 펜던트(pendant)로 보여진다. 그것은, 도 7의 장치가 픽쳐 디코더(100) 내에서 실행될 수 있다는 것이다. 도 7의 픽쳐 디코더(100)는 도 7에 따른 장치, 즉 장치(102)를 포함한다. 픽쳐 디코더(100)는, 픽쳐(104)를 복원하거나 디코딩하는데 있어, 변환 계수 블록들(10)로부터 픽쳐(104)의 블록들(106)을 차례로 픽쳐 디코더(100)로 들어가는 데이터 스트림(32)으로부터 장치(102)가 디코딩하는 복수의 변환 계수들로 재변환하도록 구성된다. 특히, 장치(102)는 블록별로 변환 블록들(10)을 처리하고, 위에서 이미 논의된대로, 상이한 크기들의 블록들(106)에 대해 공통적으로 함수(62)를 이용할 수 있다.

[0148] 픽쳐 인코더 및 디코더(60 및 100)는 각각 예측 잔류물에 변환/재변환을 적용하는 예측 코딩을 이용하도록 구성될 수 있다는 것이 주목되어야 한다. 게다가, 데이터 스트림(32)은 변환의 개별적 대상이 되는 블록들로의 분할을 픽쳐 디코더(100)에 시그널링하는, 인코딩된 서브분할 정보(subdivision information)를 포함할 수 있다.

[0150] 아래에서, 위 실시예들은 다른 몇몇 단어들로 다시 설명되고, 위 실시예들 상에서 개별적으로 전환될 수 있는 특정 관점들 상에서 더 자세한 내용들을 제공한다. 그것은, 블록-기반 이미지 및 비디오 코더들, 및 그것의 관점들처럼 계수들을 변환시키는 것에 관련된 구문 요소들(syntax elements)의 코딩에 대해 컨텍스트 모델링의 특정 방식에 관련되어 있는 위 실시예들이 설명되며 아래에서 강조된다는 것이다.

[0152] 실시예들은, 디지털 신호 프로세싱의 분야, 특히, 이미지 및 비디오 디코더들 및 인코더들에 대한 장치 및 방법에 관련될 수 있다. 특히, 블록-기반 이미지 및 비디오 코덱들에서 변환 계수들의 코딩 및 그들의 관련된 구문 요소들은 설명된 실시예들에 따라 수행될 수 있다. 그러는 한, 몇몇 실시예들은 개연성 모델링을 이용하는 엔트로피 코더를 갖는 변환 계수들에 관련된 구문 요소들의 코딩에 대한 향상된 컨텍스트 모델링을 표현하였다. 게다가, 잔여 절대 변환 계수들(remaining absolute transform coefficients)의 적응 이진화(adaptive binarization)에 이용되는 라이스 파라미터(Rice parameter)의 유도는 기호화 파라미터에 관해 위에서 설명된 것처럼 수행될 수 있다. 컨텍스트 메모리 관점에서 통합, 단순화, 우호적 병렬 처리, 및 적절한 메모리 사용은 직접 컨텍스트 모델링에 비교하여 실시예들의 이점이다.

[0154] 다른 말로, 본 발명의 실시예들은 블록-기반 이미지 및 비디오 코더들에서 변환 계수들에 코딩에 관련된 구문 요소들의 컨텍스트 모델 선택에 대한 새로운 접근을 공개한다. 게다가, 라이스 파라미터같은, 절대 변환 계수들의 잔여 값의 이진화를 제어하는, 기호화 파라미터에 대한 유도 규칙들이 설명되었다. 필수적으로, 위 실시예들은 변환 계수들의 코딩에 관련되는 구문 요소들의 모든 또는 부분에 대한 컨텍스트 모델 선택에 대한 규칙의 단순 및 공통 집합을 이용하였다.

[0156] 위에서 설명된 제1기호화 설계는 절단 단항 이진화일 수 있다. 그렇다면, coeff_significant_flag, coeff_abs_greater_1, 및 coeff_abs_greater_2 는 변환 계수의 절단 단항 이진화로부터 도출하는 제1, 제2, 제

3 빈(bin)을 형성하는 이진 구문 요소들 또는 기호들을 호출할 수 있다. 위에서 설명된 것처럼, 절단 단항 이진화는, 이는 제2레벨 인터벌(18) 내의 범위에 들어가는 변환 계수의 레벨의 경우에 라이스 코드 그 자체인 접미사가 수반될 수 있는, 접미사를 표현할 수도 있다. 추가 접미사는 0-order 처럼 Exp-Golomb 코드일 수 있고, 그런 이유로 도 2의 제1 및 제2인터벌들(16 및 18)을 따르는 추가 레벨 인터벌을 형성한다.

[0158] 위에서 설명된 것처럼, 잔여 절대 변환 계수의 적응 이진화에 대한 라이스 파라미터의 유도가 수행될 수 있고, 이는 컨텍스트 모델 선택에 대해 이용되는 것처럼 규칙들(52)의 동일 집합에 기반한다.

[0160] 스캔 순서에 관하여, 동일한 것이 위 설명과 비교하여 변화될 수 있다. 게다가, 상이한 블록 크기들 및 형태들이, 그러나 규칙들의 동일한 집합을 이용하여, 즉 동일 함수(52)를 이용하여, 도 4 및 6의 장치에 의해 뒷받침될 수 있다. 따라서, 기호화 파라미터의 유도에 대한 조화와 결합된 변환 계수들의 코딩에 관련된 구문 요소들의 컨텍스트 모델 선택에 대한 통합된 그리고 단순화된 설계가 달성될 수 있다. 이와 같이, 컨텍스트 모델 선택 및 기호화 파라미터 유도는 예를 들어, 하드와이어드(hardwired), 프로그래밍된 하드웨어 또는 소프트웨어-서브루틴일 수 있는 동일 로직(logic)을 이용할 수 있다.

[0162] 라이스 파라미터 같은, 컨텍스트 모델 선택 및 기호화 파라미터 유도를 달성하기 위해, 블록 또는 형태의 이미 코딩된 변환 계수들은 위에서 설명된 것처럼 평가될 수 있다. 이미 코딩된 변환 계수들을 평가하기 위해, (중요성 맵의 코딩에 따라 언급될 수 있는) 이진화로부터 도출하는 제1빈(first bin)인, coeff_significant_flag의 코딩에서의 구분, 및 변환 계수 레벨의 잔여 절대 값은 동일 함수(52)를 이용하여 수행된다.

[0164] 부호 정보(sign informatin)의 코딩은 삽입 방식으로 수행될 수 있고, 즉 절대 변환 계수의 코딩 후에 바로 부호를 코딩하는 것에 의해서이다. 이와 같이, 전체 변환 계수들은 하나의 스캔 패스(pass)만으로 코딩될 것이다. 대안적으로, 부호 정보는 측정 값들 $f(x)$ 가 절대 레벨 정보에만 의존하는 한 구분된 스캐닝 패스(scanning path)로 코딩될 수 있다.

[0166] 위에서 설명된 것처럼, 변환 계수들은 단일 스캔 패스로 코딩될 수 있고 또는 멀티 스캔 패스들로 코딩될 수 있다. 이는 컷오프 셋(cutoff set) c에 의해 가능하며, 그 계수 c_i 는 스캔 i에서 처리되는 변환 계수의 (제1 및 제2) 기호화의 기호 숫자를 가리킨다. 빈 컷오프 셋의 경우에, 하나의 스캔이 이용될 것이다. 기호화 파라미터의 유도 및 컨텍스트 모델 선택에 대한 향상된 결과를 가지기 위해, 컷오프 셋 c의 제1컷오프 파라미터 c_0 는 1보다 커야한다.

[0168] 컷오프 셋 c는 $c_0=1$ 및 $c_1=3$ 및 $|c|=2$와 함께 $c=\{c_0; c_1\}$로 선택될 수 있고, 여기서 c_0 는, 제1스캔에서 포함되는, 제1이진화의 빈들/기호들의 숫자를 지칭하며, $c_1=3$ 는 제1이진화의 기호들이 제2스캔에서 커버되는 기호들까지 제1이진화 내에서 기호 위치를 지칭한다. 또다른 예는, 제2스캔 패스에서 전체 블록 또는 형태에 대한 제2빈 옆에, 설계 코드 제1빈이 제1스캔 패스에서 전체 블록 또는 형태에 대한 이진화로부터 도출될 때, c_0 가 1과 같을 때, c_1 가 2와 같을 때, 등등에서 주어진다.

[0170] coeff_significant_flag 의 코딩에 대한 로컬 템플릿(56), 즉 이진화 프로세스로부터의 제1빈은, 도 1에서 보여지는 것처럼, 또는 도 9에서 보여지는 것처럼 설계될 수 있다. 통합 및 단순화에서처럼, 로컬 템플릿(56)은 모든 블록 크기들 및 형태들에 대해 이용될 수 있다. 0과 같지 않은 변환 계수를 갖는 인접한 것들의 숫자를 평가하는 것 대신에, 전체 변환 계수들은 x_i의 형태로 함수로 입력된다. 로컬 템플릿(56)은 고정될 수 있고, 즉 이전에 코딩된 변환 계수들로부터 독립적으로 그리고 스캔 인덱스 또는 현재 변환 계수의 위치로부터 독립적으로, 또는 적응적으로, 즉 이전에 코딩된 변환 계수들 및/또는 스캔 인덱스 또는 현재 변환 계수의 위치에 의존적으로 될 수 있고, 크기는 고정되거나 적응적일 수 있다. 게다가, 템플릿 크기 및 형태가 블록 또는 형태의 모든 스캔 위치들의 커버리지를 허락하면서 조정되는 경우, 이미 코딩된 변환 계수들 모두 또는 특정 한계까지의 이미 코딩된 변환 계수들 모두는 평가 프로세스에 대해 이용된다.

[0172] 예로서, 도 9는 대각 스캔(14)와 함께 8x8 변환 블록(10)에 대해 이용될 수 있는 로컬 템플릿(56)에 대한 또다른 예를 보여준다. L 은 최종 중요 스캔 위치를 나타내고 x로 표시되는 스캔 위치들은 현재 스캔 위치를 나타낸다. 다른 스캔 순서들에 대해, 로컬 템플릿은 스캔 순서(14)에 맞게 수정될 수 있다는 것을 주목하라. 예를 들어, 포워드 대각 스캔의 경우에(forward diagonal scan), 로컬 템플릿(56)은 대각선들을 따라 접힐 수 있다.

[0174] 컨텍스트 모델 선택 및 기호화 파라미터 유도는 이미 코딩된 인접 x_i의 측정으로부터 도출하는 상이한 측정 값 $f(x)$ 에 기반할 수 있다. 이 측정은 로컬 템플릿(56)에 의해 커버되는 이미 코딩된 인접한 것들을 갖는 모든 스

캔 위치들에 대해 수행된다. 로컬 템플릿(56)은 가변적인 또는 고정된 크기를 가지며 스캔 순서에 의존할 수 있다. 그러나, 템플릿 형태 및 크기는 스캔 순서에만 적용이며 값들 $f(x)$의 유도는 스캔 순서(140) 및 템플릿(56) 형태 및 크기로부터 독립적이다. 템플릿(56)의 형태 및 크기의 설정에 의해 모든 스캔 위치에 대한 블록(10)의 모든 스캔 위치들의 커버리지(coverage)가 허용된다는 것에 주목해야 하며, 현재 블록 또는 형태에서 이미 코딩된 변환 계수들 모두의 이용이 달성된다.

[0176] 이전에 언급된 것처럼, 컨텍스트 모델 인덱스들(지수들, indices)의 선택 및 기호화 파라미터의 유도는 측정 값들 $f(x)$를 이용한다. 일반적으로, 맵핑 함수들의 일반 집합은 결과 측정 값들 $f(x)$를 컨텍스트 모델 인덱스 및 특정 기호화 파라미터 상에 맵핑한다. 그것에 더하여, 변환 블록 또는 형태(10) 또는 최종 중요 스캔 위치 L 안의 현재 변환 계수의 현재 공간적 위치에 따른 추가 정보는 기호화 파라미터의 유도에 대하여 그리고 변환 계수들의 코딩에 관련된 컨텍스트 모델들의 선택에 이용된다. 공간적 위치 또는 최종 정보 및 측정으로부터 도출되는 정보는 결합될 수 있고 그래서 특정 가중(weighting)가 가능하다. 측정 및 유도 프로세스 후에, 모든 파라미터들(컨텍스트 모델 지수들, 기호화 파라미터)은 특정 제한까지 변환 계수 또는 전체 변환 계수 레벨의 코딩에 대해 이용가능하다.

[0178] 본 발명의 구성 예로서, 컷오프 셋 크기는 비어있다. 이는, 각 변환 계수가 스캔 순서를 따라 다음 변환 순서들을 처리하기 전에 완전히 전송된다는 것을 의미한다.

[0180] 측정 값들 $f(x)$은 로컬 템플릿(56)에 의해 커버되는 이미 코딩된 인접한 것들 x_i의 측정으로부터 도출될 수 있다는 것이다. 특정 맵핑 함수 $f_t(\boldsymbol{x})$ 는 라이스 파라미터 및 컨텍스트 모델을 선택하는데 이용되는 측정 값에 입력 벡터를 맵핑한다. 입력 벡터 x는 삽입 설계에 의존하고 로컬 템플릿(56)에 의해 커버되는 인접한 것들의 변환 계수 값들 x_i로 구성될 수 있다. 예를 들어, 만약 컷오프 셋 c 이 비어있고 부호가 구분 스캔 패스로 코딩되는 경우, 벡터 x는 절대 변환 계수들 x_i 에 의해서만 구성된다. 일반적으로, 입력 벡터 x의 값은 부호를 가지거나 부호를 가지지 않을 수 있다. 맵핑 함수는 d 의 차원의 입력 벡터 x를 갖고 다음에 따라 공식화될 수 있다(주어진 t는 일정한 입력).

[0181] $$f_t(\boldsymbol{x}) = \sum_{i=1}^{i=d} w_i \cdot g_t(x_i) \cdot \delta(x_i, t)$$

[0183] 더 특정적으로, 맵핑 함수 $f_t(\boldsymbol{x})$ 는 d의 차원의 입력 벡터 x에 따라 정의될 수 있다(t는 일정한 입력).

[0184] $$f_t(\boldsymbol{x}) = \sum_{i=1}^{i=d} w_i \cdot (|x_i| - t) \cdot \delta(x_i, t)$$

[0186] 그것은, $g_t(x_i)$ 은 $(|x_i| - t)$ 일 수 있다는 것이다. 후자의 공식에서, 함수 δ는 다음에 따라 정의된다.

[0187] $$\delta(x,t) = \begin{cases} 1 & |x| \geq t \\ 0 & |x| < t \end{cases} \delta(x,t) = \begin{cases} 1 & |x| \geq t \\ 0 & |x| < t \end{cases} \quad (1)$$

[0188] 측정 값의 또다른 종류는 다음에 따라 정의되는 특정 값 t보다 크거나 작은 인접 절대 변환 계수 레벨의 숫자이다:

[0189] $$f_t(\boldsymbol{x}) = \sum_{i=0}^{i=d} w_i \cdot \delta(x_i, t) \quad f_t(\boldsymbol{x}) = \sum_{i=0}^{i=d} w_i \cdot \delta(x_i, t)$$

[0190] 측정 값들 두 종류에 대해, 특정 인접한 것의 중요성을 제어하는 추가 가중 인수가 가능하다. 예를 들어, 가중 인수 w_i 는 더 큰 공간적 거리와 인접한 것에 대해서보다 더 짧은 공간적 거리의 인접한 것들에 대해 더 높다. 게다가 가중은 1로 모든 w_i가 설정될 때 무시된다.

[0192] 본 발명의 예시 구성에 따라, f_0, f_1, f_2 및 f_3 는 (1)에서 정의되는 δ(x_i) 및 *{0, 1, 2, 3}의* 각 t를 갖는 측정 값이다. 이 예에 대해, 라이스 파라미터에 대한 f_3, 제3빈에 대한 f_2, 제2빈에 대한 f_1, 제1빈의 컨텍스트 인덱스의 유도에 대해 이용된다. 또다른 예시 구성에서, f_0 는 제1빈의 컨텍스트 모델 선택에 대해 이용되고, 반면 f_1 는 제2, 제3빈, 및 라이스 파라미터의 컨텍스트 모델 선택에 대해 취해진다. 여기서, 라이스 파라미터는 다른 기호화 파라미터들에 대한 표현(대표, representative)로서 기능한다. 기호화 파라미터 및 엔트로피 코딩에

있어서 및 기호화 모든 구문 요소들 또는 빈 인덱스들에 대한 컨텍스트 모델 선택은 측정 값들 $f(x)$를 활용하여 동일 로직을 이용한다. 일반적으로, 특정 측정 값 f(x)는 기호화 파라미터 또는 컨텍스트 모델 인덱스에 또다른 맵핑 함수 $g(x,\boldsymbol{n})$에 의해 맵핑된다. 특정 맵핑 함수는 입력 벡터 n의 차원으로서 d 를 따라 정의된다.

[0193] $$g(\boldsymbol{x}) = \sum_{i=1}^{i=d} \delta'(x, n_i)$$

[0194] 이러한 맵핑에 대해, 함수 $\delta(x,\boldsymbol{n})$는 다음에 따라 정의될 수 있다.

[0195] $$\delta'(x,n) = \begin{cases} 1 & x > n \\ 0 & x \leq n \end{cases}$$

[0196] 입력 벡터 n의 차원 d 및 벡터 n의 값은 가변적일 수 있고 구문 요소 또는 빈 인덱스에 의존할 수 있다. 게다가, 변환 블록 또는 형상 내의 공간적 위치는 선택된 컨텍스트 모델 인덱스를 추가 또는 감산 (또는 이동) 시키도록 이용될 수 있다.

[0198] 동일한 것을 코딩/디코딩할 때 변환 계수를 스캐닝하는 제1스캔 위치는, DC 로부터 가장 높은 주파수까지를 가리키는 도 1의 스캔 방향을 적용할 때 최종 스캔 위치 L 이 될 수 있다. 그것은, 동일한 것을 코딩/디코딩하기 위해 계수들을 검색하기 위한 스캔의 제1스캔이 적어도, 계수 L 로부터 DC 까지를 가리킬 수 있다는 것이다. 이러한 스캔 위치 L에 대해, 제1빈 인덱스는 이 스캔 위치가 0과 동일하지 않은 변환 계수로 구성된다는 것이 이미 시그널링된 최종 정보로서 무시될 수 있다. 이러한 스캔 위치에 대해, 개변 컨텍스트 모델 인덱스는 변환 계수의 이진화로부터 도출하는 제2 및 제3빈의 코딩에 이용될 수 있다.

[0200] 본 발명의 예시 구성으로, 결과 측정 값 f_0 는 입력 벡터 $\boldsymbol{n}=\{1,2,3,4,5\}$ 와 함께 입력으로서 이용되고, 결과 값은 제1빈에 대해 컨텍스트 모델 인덱스이다. 0과 동일한 측정 값의 경우에, 컨텍스트 인덱스는 0이라는 것에 주목하자. 동일 설계는 측정 값 f_1 및 입력 벡터 $\boldsymbol{n}=\{1,2,3,4\}$ 와 함께 적용되고 결과 값은 이진화의 제2 및 제3빈에 대한 컨텍스트 모델 인덱스이다. 라이스 파라미터에 대해, f_3 및 $\boldsymbol{n}=\{0,5,19\}$ 이 이용된다. 최대 라이스 파라미터는 3이고 최신 기술과 비교하여 최대 라이스 파라미터에는 어떠한 변화가 본 발명에 의해 이루어지지 않는다는 것에 주목하라. 대안적으로, f_1 는 라이스 파라미터를 유도하도록 이용될 수 있다. 그러한 구성에 대해, 입력 벡터는 $\boldsymbol{n}=\{3,9,21\}$로 수정되어야 한다. 규칙(rules)의 기본적 집합은 모든 구문 요소 또는 빈 인덱스들 및 라이스 파라미터에 대해 동일하고, 다만 파라미터들 또는 임계 집합들 (입력 벡터 $\boldsymbol{n}$)이 다르다는 것에 주목하라. 게다가, 현재 스캔 위치의 대각선에 의존하여, 컨텍스트 모델 인덱스는 특정 량을 더하거나 감산하는 것에 의해 이전에 언급된 것처럼 수정될 수 있다. 그것에 대한 동일한 설명은 또다른 분리 컨텍스트 모델 집합(disjoint context model set)의 선택이다. 예시 실행에 있어서, 만약 현재 스캔 위치가 첫번째 두 대각선 상에 있는 경우 제1빈에 대한 결과 컨텍스트 모델 인덱스는 $2*/ctx0/$ 에 의해 이동된다. 현재 스캔 위치가 제3 및 제4대각선 상에 있는 경우, 제1빈에 대한 컨텍스트 모델 인덱스는 $/ctx0/$에 의해 이동되고, $/ctx0/$ 는 분리 컨텍스트 모델 집합들을 도출하는 측정 값들에 기반하는 유도(derivation)으로부터 도출되는 최대 컨텍스트 모델들의 숫자이다. 이 개념은 예시 실행에 대해서만 루마 플레인(luma planes)에 대해 이용되고, 반면 추가 오프셋(further offset)이 컨텍스트 희석을 피하는 채도의 경우에 더해지지 않는다(즉 충분하지 않은 빈들이 적응 컨텍스트 모델과 코딩되며 통계는 컨텍스트 모델에 의해 추적되지 못한다). 동일 테크닉이 제2 및 제3빈의 컨텍스트 모델 인덱스에 적용될 수 있다. 여기서, 제안된 발명의 예시 구성에서, 임계 대각선은 3 및 10이다. 다시, 이 테크닉은 루마 신호에만 적용된다. 이러한 테크닉을 채도 신호등에 확장하는 것 또한 가능하다는 것에 주목하라. 게다가, 대각선(diagonals)들에 의존하여 추가 인덱스 오프셋이 다음에 따라 공식화될 수 있다는 것에 주목하라.

[0202] $$ctx_{offset} = d_j * idx_{inc}$$

[0204] 이 공식에서 d_j 는 현재 스캔 위치의 대각선에 대한 무게를 나타내고 idx_{inc}는 스텝 크기(step size)를 나타낸다. 게다가, 오프셋 인덱스가 실질적 실시예에 대해 반전(inverted) 될 수 있다는 것에 주목하라. 언급된 예시 실행에 대해, 현재 스캔 위치가 제1 및 제2 대각선에 있는 경우 반전(inversion)은 추가 인덱스를 0으로 설정할 것이고, 제3 및 제4 대각선에 대해 $/ctx0/$ 에 의해 이동되고 그렇지 않은 경우 $2*/ctx0/$ 이다. 주어진 공식을 이용하는 것에 의해, d_0 및 d_1 내지 2, d_3 및 d_4 내지 1 및 모든 잔여 대각선 인수들이 0으로 설정될 때, 예시 구성에 대한 동일한 거동이 달성된다.

[0206] 컨텍스트 모델 인덱스가 상이한 블록 크기들 또는 플레인(plane) 타입들(예를 들어, 루마 및 채도)에 대해 동일하다 하더라도, 기본 컨텍스트 모델 인덱스는 컨텍스트 모델들의 상이한 집합들을 도출하며 다를 수 있다. 예를 들어, 루마에서 8x8 보다 큰 블록 크기들 동일 기본 인덱스가 이용될 수 있고, 반면 기본 인덱스는 루마에서 4x4 및 8x8 에 대해 상이할 수 있다. 컨텍스트 모델들의 의미있는 숫자를 가지기 위해, 기본 인덱스는, 그러나 상이한 방법으로 그룹화(grouped) 될 수 있다.

[0208] 예시 구성으로, 4x4 블록들에 대한 에 대한 컨텍스트 모델들 및 잔여 블록들은 루마에서 상이할 수 있고, 반면 동일 기본 인덱스는 채도 신호에 대해 이용될 수 있다. 또다른 예에서, 동일 기본 인덱스는 양쪽 루마 및 채도 신호들에 대해 이용될 수 있고, 반면 루마 및 채도에 대한 컨텍스트 모델들은 상이하다. 게다가, 제2 및 제3빈들에 대한 컨텍스트 모델들은 컨텍스트 메모리의 더 작은 숫자를 도출하면서 그룹화될 수 있다. 제2 및 제3빈에 대한 컨텍스트 모델 인덱스 유도가 동일한 경우, 동일 컨텍스트 모델은 제2 및 제3빈을 전송하도록 이용될 수 있다. 기본 인덱스 그룹핑 및 가중의 올바른 조합에 의해, 컨텍스트 모델들의 의미있는 숫자가 컨텍스트 메모리의 절약을 도출하면서 달성될 수 있다.

[0210] 본 발명의 바람직한 실시예에서, 컷오프 집합 c 는 비어있다. 그것은, 단지 하나의 스캔이 이용된다는 것이다. 이러한 바람직한 실시예에서, 부호 정보는 동일 스캔 패스를 이용하여 삽입될 수 있고(인터리브드, interleaved) 또는 구분 스캔 패스(separate scan pass)로 코딩될 수 있다. 또다른 바람직한 실시예에서, 집합 크기 c 는 1 및 c_0 에 동일하고, 컷오프 집합 c의 제1 및 유일한 값은 3과 동일하다. 이는 두개의 스캔을 이용하여 위에서 설명된 예에 대응한다. 바람직한 실시예에서, 컨텍스트 모델 선택은 절단 단항 이진화를 도출하면서 모든 세개의 빈들에 대해 수행될 수 있고 반면 라이스 파라미터 선택같은 기호화 파라미터 유도는 동일 함수(52)를 이용하여 수행될 수 있다.

[0212] 바람직한 실시예에서, 로컬 템플릿의 크기는 5이다. 로컬 템플릿의 크기는 4일 수 있다. 이 바람직한 실시예에서, 두개의 수직 방향으로 2의 공간적 거리를 갖는 인접한 것들이 도 8과 비교하여 제거될 수 있다. 더 바람직한 실시예에서, 템플릿 크기는 적응적이며 스캔 순서로 조정된다. 이 바람직한 실시예에 대해, 이전에 처리 단계에서 코딩된 인접한 것들은 도 1 및 8의 경우에서처럼 그대로 템플릿에 포함되지 않는다. 이렇게 함으로써, 의존도 또는 대기 시간(latency)은 짧아지며, 더 높은 프로세싱 순서를 도출한다. 더 바람직한 실시예에서, 템플릿 크기 및 형태는 충분히 크도록 조정된다 (예를 들어, 현재 블록 또는 형태의 동일 블록 또는 형태 크기) 또다른 바람직한 실시예에서, 두개의 로컬 템플릿들이 이용될 수 있고 그것들은 가중 인수에 의해 결합될 수 있다. 이 바람직한 실시예에 대해, 로컬 템플릿들은 크기 및 형태에서 다를 수 있다.

[0214] 바람직한 실시예에서, f_0 는 그리고 제2빈(bin), 제3빈, 및 라이스 파라미터에 대한 f_1 및 제1빈에 대한 컨텍스트 모델 인덱스를 선택하는데 이용될 수 있다. 이러한 바람직한 실시예에서, 입력 벡터 $\mathbf{n}=\{0,1,2,3,4,5\}$ 는 6개의 컨텍스트 모델들을 도출한다. 제2 및 제3빈 인덱스에 대한 입력 벡터 n은 동일(same) 및 $\mathbf{n}=\{0,1,2,3,4\}$일 수 있고, 반면 라이스 파라미터에 대한 입력 벡터 n은 $\mathbf{n}=\{3,9,21\}$일 수 있다. 게다가, 바람직한 실시예에서, 구분 컨텍스트 집합들이 이용되는 것들 내의 변형 블록의 미리-언급된 주파수 부분들은 대각선 (래스터(raster)) 스캔의 대각선들 (또는 라인들)의 분리 집합들에 의해 형성될 수 있다. 예를 들어, 상이한 컨텍스트 기본 오프셋 숫자들이 제1 및 제2대각선들에 대해, DC 구성요소로부터 보일 때 제2 및 제3대각선들 및 제4 및 제5대각선들에 대해, 존재할 수 있고, 이러한 대각선들에서 계수들에 대한 컨텍스트 선택은 컨텍스트들의 분리 집합들 내에서 일어난다. 제1대각선이 하나(one)라는데 주목하자. 제 및 제3빈 인덱스에 대해, [0, 2] 사이의 범위에 존재하는 대각선들은 2의 가중 인수를 가지며, [3, 9] 사이의 범위에 존재하는 대각선들은 1의 가중 인수를 갖는다. 이러한 추가적 오프셋들은 루마 신호의 경우에 이용되며, 채도에 대한 가중 인수들은 모두 0과 동일하다. 이 바람직한 실시예에 대해서도, 최종 중요 스캔 위치인, 제1스캔 위치의 제2 및 제3빈 인덱스에 대한 컨텍스트 모델은, 잔여 컨텍스트 모델들로부터 구분된다. 이는 측정 프로세스가 구분 컨텍스트 모델을 결코 선택하지 않는다는 것을 의미한다.

[0216] 바람직한 실시예에서, 4x4 루마 블록들 또는 형태는 제1빈에 대해 컨텍스트의 단일 집합을 이용하며, 잔여 블록 크기들 또는 형태에 대한 컨텍스트 모델들은 동일하다. 이 바람직한 실시예에서, 블록 크기 또는 채도 신호에 대한 형태 사이의 구분이 없다. 본 발명의 또다른 바람직한 실시예에서, 모든 블록 크기들 및 형태에 대한 기본 베이스 인덱스 또는 컨텍스트 모델 집합들을 도출하는 블록 크기들 또는 형태 사이의 구분이 없다. 양쪽 바람직한 실시예에서, 컨텍스트 모델들의 상이한 집합들이 루마 및 채도 신호들에 대해 이용된다는 것에 주목하라.

[0218] 아래에서, 위 실시예에 따라 수정된 라이스 파라미터 이진화를 이용하지만, 컨텍스트 적응 엔트로피 코딩 없는, 실시예가 보여진다. 이 대안적 코딩 설계에 따라, 라이스 이진화 설계만이 이용된다(선택적으로, Exp-Golomb 접

미사의 추가와 함께). 이와 같이, 적응 컨텍스트 모델은 변환 계수를 코딩하는데 필요하지 않다. 대안적 코딩 설계에 대해, 라이스 파라미터 유도는 위 실시예들에 대해서처럼 동일 규칙(룰, rule)을 이용한다.

[0220] 다른말로, 코딩 파이프라인(pipeline)에서 대기 시간을 향상시키기 위해 그리고 컨텍스트 메모리 및 복잡성을 감소시키기 위해, 규칙 또는 로직의 동일 집합에 기반한 대안적 코딩 설계가 설명된다. 이러한 대안적 코딩 설계에 대해, 이진화로부터 첫번째 세개의 빈들에 대한 컨텍스트 모델 선택은 불가능하고, 즉 제1기호화 설계인, 절단 단항 이진화로부터 도출되는 첫번째 세개의 빈들,은 고정된 동일 개연성을 가지고 코딩될 수 있다(즉 0.5의 개연성을 가짐). 대안적으로, 절단 단항 이진화 설계는 생략되며 이진화 설계의 인터벌 경계(인터벌 바운드, interval bounds)가 조정된다. 이 활용에서, 라이스 인터벌의 왼쪽 경계, 즉 인터벌(18)은 3 대신에 0이다 (사라지는 인터벌(16)과 함께). 이 활용에서 오른쪽/위쪽 경계는 수정될 수 있고 또는 3에 의해 감산될 수 있다. 라이스 파라미터의 유도는 입력 벡터 n의 관점에서 그리고 측정 값의 관점에서 수정될 수 있다.

[0222] 이와 같이, 바로-설명된 수정 실시예들에 따라, 데이터 스트림(32)로부터, 각각이 변환 계수 레벨을 갖는, 상이한 변환 블록들의 복수의 변환 계수들을 디코딩하는 장치는, 도 10과 관련하여 설명된다.

[0224] 도 10의 장치는, 현재 변환 계수에 대한 데이터 스트림(32)로부터 기호 시퀀스(122) 또는 기호들의 집합을 추출하도록 구성된다. 추출은 도 7의 추출기(84)와 관련하여 위에서 설명된대로 수행된다.

[0226] 역기호화기(124)는 기호화 파라미터에 따라 파라미터화 가능한 기호화 설계에 따라 현재 변환 계수에 대해 변환 계수 레벨 상에 기호들의 집합(122)를 맵핑하도록 구성된다. 상기 맵핑은 라이스 이진화처럼 파라미터화 가능한 기호화 설계만을 이용할 수 있거나, 현재 변환 계수의 전체 기호화의 접두사 또는 접미사처럼 파라미터화 가능한 기호화를 이용할 수 있다. 도 2의 경우, 예를 들면, 파라미터화 가능한 기호화 설계, 즉 두번째 것은, 제1기호화 설계의 기호 시퀀스에 관련된 접미사를 형성하였다.

[0228] 예를 더 들어보면, 도 11a 및 b를 참고하라. 도 11a에 따라, 변환 계수의 인터벌 범위(20)는 세개의 인터벌들(16, 18 및 126)으로 서브분할되고, 각 더 낮은 인터벌의 개별 최대 레벨에서 서로 중첩하고 인터벌 범위(20)을 함께 커버한다. 계수 레벨 x가 가장 높은 인터벌(126) 내에 있는 경우, 전체 기호화는 인터벌(16) 내의 레벨을 기호화하는 제1기호화 설계(128)의 기호 시퀀스(44)의 조합이며, 기호 시퀀스는 접두사를 형성하고 이는 제1접미사가 뒤따르며, 즉 제2기호화 설계(130)의 기호 시퀀스(42)는 인터벌(18) 내에서 레벨들을 기호화하며, 더 나아가 제2의 접미사가 뒤따르고, 즉 제3기호화 설계(134)의 기호 시퀀스(132)는 인터벌(126) 내에서 레벨들을 기호화한다. 후자의 것은 순서(order) 0처럼 Exp-Golomb 코드일 수 있다. 만약 계수 레벨 x가 중간 인터벌(18)에 있는 경우 (그러나 인터벌(126) 내에 있는 것이 아니라면), 전체 기호화는 제1접미사(42)가 뒤따르는 접두사(44)의 조합이다. 만약 계수 레벨 x가 가장 낮은 인터벌(16) 내에 있는 경우 (그러나 인터벌(18) 내가 아니라면), 전체 기호화는 단지 접두사 (44)로 구성된다. 전체 기호화는 동일한 것이 접미사 없이 구성된다. 기호화 없이, 도 11a 에 따른 기호화는 도 2 중 하나와 대응할 수 있다. 제3기호화 설계(134)는 Golomb-Rice 이진화일 수 있다. 비록 동일한 것이 첫번째(128)일 수도 있지만, 제2기호화 설계(130)는 파라미터화 가능한 것을 형성할 수 있다.

[0230] 대안적인 전체 기호화는 도 1에서 보여진다. 여기서, 단지 두개의 기호화 설계들이 결합된다. 도 11a와 비교하여, 제1기호화 설계가 빼내진다. 설계(130)의 인터벌(138) 또는 설계(134)의 인터벌(136) 내의 x에 의존하여, x의 기호화는 접두사(140) 및 접미사(142), 또는 단지 접두사(140)를 포함한다.

[0232] 게다가, 도 10의 장치는 역기호화기(124)의 파라미터 입력 및 역기호화기의 출력 사이에 연결되는 기호화 파라미터 결정기(144)를 포함한다. (여태까지 역기호화된/처리된/디코딩된 부분들 또는 역기호화된 부분들로부터 유도가능 하다면) 결정기(144)는, 함수(52)를 통해, 이전에 처리된 변환 계수들 상에 의존하는 현재 변환 계수에 대해 기호화 파라미터(46)를 결정하도록 구성된다.

[0234] 추출기(120), 역기호화기(124) 및 기호화 파라미터 결정기(144)는 위에서 설명된대로 상이한 변환 블록들의 변환 계수들을 순차적으로 처리하도록 구성된다. 그것은, 스캔(140)이 변환 블록(10) 내에 반대 방향으로 검색될 수 있다는 것이다. 몇몇 스캔들은 예를 들어, 상이한 기호화 부분들, 즉 접두사 및 접미사(들)처럼 이용될 수 있다.

[0236] 함수 파라미터는 현재 변환 계수의 변환 블록 및/또는 주파수 부분의 정보 구성요소 타입, 현재 변환 계수의 변환 블록의 크기에 의존하여 변화하며, 현재 변환 계수는 변환 블록 내에 위치된다.

[0238] 상기 장치는 한편으로는 이전에 디코딩된 변환 계수들 사이의 관계를 정의하는 함수가, 위에서 설명된 함수인,

$g(f(\mathbf{x}))$으로 구성될 수 있다.

[0240] 위에서 설명된대로, 현재 변환 계수에 관련된 관련 공간적 배치에 의존하여 이전에 처리된 계수들의 공간적 결정이 이용될 수 있다.

[0242] 상기 장치는, 추출기(120)가 고정된 개연성 분포를 이용하여 엔트로피 디코딩을 이용하거나 직접 데이터 스트림으로부터 기호들의 집합을 추출하도록 구성될 수 있기 때문에, 매우 쉽고 빠르게 작동할 수 있다.

[0244] 파라미터화 가능한 기호화 설계는 기호들의 집합이 라이스 코드(Rice code)이고, 기호화 파라미터는 라이스 파라미터일 수 있다.

[0246] 다른 말로, 기호들의 집합이 44 및 132, 또는 142 처럼 현재 변환 계수의 전체 기호화의 다른 부분들에 관한 접두사 또는 접미사를 나타내도록, 역기호화기(124)는 변환 계수들의 범위 인터벌로부터 18 또는 138 같은 레벨 인터벌에 기호화 설계를 제한하도록 구성될 수 있다. 다른 기호들처럼, 동일한 것이 고정된 개연성 분포를 이용하는 엔트로피 디코딩을 이용하여 또는 직접 데이터 스트림으로부터 추출될 수도 있지만, 도 1 내지 9는 컨텍스트 적응을 이용하는 엔트로피 코딩이 이용될 수도 있다는 것을 보여주었다.

[0248] 도 10의 장치는 도 8의 픽쳐 디코더(102)에서 장치(102)로서 이용된다.

[0250] 완전성을 위해, 도 12는, 각각이 변환 계수 레벨을 갖는, 상이한 변환 블록들의 복수의 변환 계수들을, 도 10의 장치에 맞는, 데이터 스트림(32)으로 코딩하는 장치를 보여준다.

[0252] 도 12의 장치는 기호들의 집합 또는 기호 시퀀스 상에, 기호화 파라미터에 따라 파라미터화 가능한 기호화 설계에 따른 현재 변환 계수에 대한 변환 계수 레벨을 맵핑하도록 구성되는 기호화기(150, symbolizer)를 포함한다.

[0254] 삽입기(154)는 데이터 스트림으로 현재 변환 계수에 대한 기호들의 집합을 삽입하도록 구성된다.

[0256] 기호화 파라미터 결정기(156)는, 함수 파라미터를 통해 파라미터화 가능한 함수(52)를 통해, 이전에 처리된 변환 계수들 상에, 의존하는 현재 변환 계수에 대해 기호화 파라미터(46)를 결정하도록 구성되며, 이런 이유로, 기호화기(150)의 입력 및 출력 사이에 대안적으로, 또는 기호화기(150)의 파라미터 입력 및 삽입기(152)의 출력 사이에 연결될 수 있다.

[0258] 삽입기(154), 기호화기(150) 및 기호화 파라미터 결정기(156)는 상이한 변환 블록들의 변환 계수들을 순차적으로 처리하도록 구성될 수 있고, 함수 파라미터는 현재 변환 계수의 변환 블록 및/또는 주파수 부분의 정보 구성 요소 타입, 현재 변환 계수의 변환 블록의 크기에 의존하여 변할 수 있고, 현재 변환 계수는 변환 블록 내에 위치한다.

[0260] 도 10의 장치를 디코딩하는 것과 관련하여 설명된 것처럼, 도 12의 장치는 한편으로는 이전에 디코딩된 변환 계수들 및 다른 한편으로는 기호화 파라미터 사이의 관계를 정의하는 함수가 $g(f(\mathbf{x}))$이며, 이전에 처리된 변환 계수들이 현재 변환 계수에 관련된 상대적 공간적 배치에 의존하여 공간적으로 결정될 수 있도록 구성될 수 있다. 삽입기는 고정된 개연성 분포를 이용하는 엔트로피 인코딩을 이용하거나 직접 데이터 스트림으로 기호들의 집합을 삽입하도록 구성될 수 있고, 기호화 설계는 기호의 집합이 라이스 코드이고, 기호화 파라미터가 라이스 파라미터일 수 있다. 기호들의 집합이 현재 변환 계수의 전체 기호화의 다른 부분에 관련된 접두사 또는 접미사를 표현하도록 기호화기는 변환 계수들의 범위 인터벌(20)로부터의 레벨 인터벌에 기호화 설계를 제한하도록 구성될 수 있다.

[0262] 위에서 언급된 것처럼, 도 1 내지 12의 바람직한 실시예에서, 첫번째 세 빈들(first three bins)은 도 1 내지 9의 실시예와 비교하여 이용불가하다. 이 바람직한 실시예에 대해, 절단 단항 이진화(128)로부터 도출되는 빈들은 고정된 개연성 0.5로 코딩된다. 더 바람직한 실시예에서, 절단 단항 이진화(128)는 도 11b에서 보여지는 것처럼 생략되고 라이스 인터벌에 대한 경계는 최신 기술에서처럼 동일 인터벌 범위에서 도출되며 조정된다(즉 왼쪽 및 오른쪽 경계 빼기 3). 이 바람직한 실시예에 대해, 라이스 파라미터 유도 규칙은 도 1 내지 9의 실시예와 비교하여 수정된다. 평가 값에 따라 f_1 를 이용하는 대신에, 예를 들어, f_0 가 이용될 수 있다. 게다가, 입력 벡터는 $\mathbf{n}=\{4,10,22\}$로 조정될 수 있다. 이 아래에서 설명되는 추가 실시예는, 한편으로는 컨텍스트 선택/의존도에 대한 다른 한편으로는 기호화 파라미터 결정에 대한 사실상 상이한 템플릿들을 갖는 가능성을 도시한다. 이것은, 계수 x_i의 템플릿은 기호화 파라미터 결정 및 컨텍스트 선택/의존도 양쪽에 대해 동일한 것이 남아있다는 것이고, 다만 f(x) 에 영향을 미치는데 참여하는 계수들 x_i는 적절한 설정 w_i 에 의해 기호화 파라미터 및 컨

텍스트 선택/의존도사이에서 다르게 효과적으로 렌더링된다: 가중 w_i이 0인 모든 계수들 x_i는 따라서 f(x)에 영향을 미치지 않고, w_i가 0인 템플릿의 부분들을 설계하며, 이는 한편으로는 컨텍스트 선택/의존 및 다른 한편으로 기호화 파라미터 결정 사이에서 다르며, 컨텍스트 선택/의존 및 기호화 파라미터 결정에 대한 상이한 "효과적 템플릿"을 효과적으로 도출한다. 다른 말로, 컨텍스트 선택/의존 및 기호화 파라미터 결정 중 하나에 대한 특정 템플릿 위치 i에 대해 몇몇 w_i를 0으로 설정하는 것에 의해, 이러한 특정 템플릿 위치들 i에서 w_i를 컨텍스트 선택/의존 및 기호화 파라미터 결정에 대한 비-제로 값으로 설정하는 동안, 먼저 언급된 것의 템플릿 컨텍스트 선택/의존 및 기호화 파라미터는 컨텍스트 선택/의존도 및 기호화 파라미터 결정 중 후자(latter)의 템플릿보다 효과적으로 작다. 다시, 이미 위에서 나타내진것처럼, 템플릿은, 예를 들어 현재 코딩된 변환 계수의 위치의 위치에 무관하게, 블록의 전체 변환 계수들을 포함할 수 있다.

[0265] *예를 들어, 16x16 변환 계수 (12)의 배치로 예시적으로 구성되는, 변환 계수 블록(10)을 보여주는 도 13을 보라. 변환 계수 블록(10)은 각각 4x4 변환 계수들 (12)의 서브-블록들(200)로 서브분할된다. 서브-블록들(200)은 4x4 배치에서 정기적으로 배치된다. 본 실시예에 따라, 변환 계수 블록(10)을 코딩하는데 있어, 중요성 맵(significance map)은 데이터 스트림(32)내에서 코딩되며, 중요성 맵은 중요한 변환 계수 레벨들(12), 즉 0과 동일하지 않은 변환 계수 레벨들,의 위치를 나타낸다. 이후, 변환 계수 레벨에서 이러한 중요 변환 계수들의 하나를 빼는 것은 데이터 스트림 내에서 코딩될 수 있다. 나중의 변환 계수 레벨들의 코딩은 위에서 설명되는대로 수행될 수 있고, 즉 기호화 파라미터를 결정하고 컨텍스트를 선택하는 일반 파라미터화 가능한 함수를 이용하는 가변 길이 코딩 설계 및 믹스된 컨텍스트 적응 엔트로피 코딩에 의해서이다. 특정 스캔 순서는 중요한 변환 계수들을 연속화하거나 (serialize) 순서화하기 위해 (order) 이용될 수 있다. 그러한 스캔 순서의 하나의 예는 도13에서 도시된다: 각 서브-블록(200) 내에서, 서브-블록들(200)은 가장 높은 주파수(오른쪽 아래) 에서 DC (왼쪽 위)까지 스캔되며, 이는 서브-블록 순서로 다음 서브-블록의 변환 계수들이 방문되기 전에 변환 계수들(12)이 스캔된다. 이는 서브-블록 스캔을 나타내는 화살표(202)에 의해 도시되며, 204는 실제 계수 스캔의 부분을 도시한다. 스캔 인덱스는 서브-블록들 내에서 변환 계수들(12) 및/또는 서브-블록들(200)을 스캐닝하는데 몇몇 스캔 순서들 중에서의 선택을 허용하기 위해 데이터 스트림(32) 내에서 전송될 수 있다. 도 13에서, 대각선 스캔은 서브-블록 스캔(202) 및 각 서브-블록 내의 변환 계수(12)의 스캔 모두에 대해 설명된다. 따라서, 디코더에서, 중요성 맵이 디코딩되며, 방금 언급된 스캔 순서를 이용하여 그리고 파라미터화 가능한 함수를 이용하여 위 실시예들을 이용하여 중요성 변환 계수들의 변환 계수 레벨들이 디코딩될 것이다. 아래에서 더 자세히 나타나는 설명에서, xS 및 yS 는, 현재 코딩된/디코딩된 변환 계수가 위치되는, 즉 블록(10)의 위쪽 왼쪽 코너인, DC 위치로부터 측정된 서브-블록 열 및 서브-블록 줄을 나타낸다. xP 및 yP 는 현재 서브-블록 (xS, yS)의 위쪽 왼쪽 코너 (DC 계수 위치)로부터 측정되는 현재 코딩된/디코딩된 변환 계수의 위치를 나타낸다. 이는 오른쪽 위 서브-블록(200)에 대해 도 13에서 도시된다. xC 및 yC는 DC 포지션으로부터 변환 계수들에서 측정되는 현재 디코딩된/코딩된 변환 계수들 위치를 나타낸다. 게다가, 도 13에서 블록(10)의 블록 크기에 따라, 즉 16x16은, 설명의 목적만을 위해 선택되었으며, 아래에서 더 설명되는 실시예는 이차(quadratic)로 가정되는, 블록(10)의 크기를 나타내는 파라미터로 log2TrafoSize 를 이용한다. log2TrafoSize 는 블록(10)의 변환 계수들의 각 열 내에서 변환 계수들의 숫자를 이차 로그를 나타내며, 즉 변환 계수들에서 측정되는 블록(10)의 모서리의 길이의 $\log_2$ 이다. CtxIdxInc는 최종적으로 컨텍스트를 선택한다. 게다가, 아래에서 설명되는 특정 실시예에서, 앞에서 언급된 중요성 맵은 coded_sub_block_flag를 시그널링하도록 가정되고, 즉 이진화 구문 요소 또는 플래그이며, 시그널링하기 위한 블록(10)의 서브-블록들(200)에 대해서이며, 서브-블록 방향으로, 각 서브-블록(200) 내에서 어떠한 중요성 변환 계수가 위치되는지 아닌지, 즉 중요하지 않은 변환 계수들이 각 서브-블록(200) 내에 위치하는지 여부를 시그널링하도록 가정된다. 만약 플래그가 0이라면, 중요하지 않은 변환 계수들은 각 서브-블록 내에 위치된다.

[0267] 이와 같이, 이 실시예에 따라, significant_coeff_flag 의 컨텍스트를 선택하기 위해 다음이 컨텍스트 적응 엔트로피 디코더/인코더에 의해 수행되며, 즉 플래그는, 각 계수가 중요한지 여부에 대해, 즉 비-제로(non-zero)인지 아닌지에 대해, 각 서브-블록(200)이 비-제로 변환 계수들을 포함하는지를 coded_sub_block_flag 가 시그널링하는 서브-블록의 특정 변환 계수에 대한 신호들 및 중요성 맵의 부분이다.

[0269] 이 프로세스에의 입력들은 컬러 구성요소 인덱스 cIdx, 현재 계수 스캔 위치 (xC, yC), 스캔 순서 인덱스 scanIdx, 변환 블록 크기 log2TrafoSize 이다. 이 프로세스의 출력은 ctxIdxInc 이다. 변수 sigCtx 는 현재 위치 (xC, yC), 컬러 구성요소 인덱스 cIdx, 그구구문 요소 coded_sub_block_flag의 이전에 디코딩된 빈들 및 변환 블록 크기에 의존한다. sigCtx의 유도에 대해, 다음에 적용된다.

[0271] - 만약 log2TrafoSize 이 2와 동일하다면, sigCtx 는 다음에 따라 표(테이블) 1에서 특정된 ctxIdxMap[ ] 를 이용하여 유도된다.

[0272] sigCtx = ctxIdxMap[(yC << 2) + xC]

[0273] - 그렇지 않다면, 만약 xC + yC가 0과 같다면, sigCtx는 다음에 따라 유도된다.

[0274] sigCtx = 0

[0275] - 그렇지 않다면, sigCtx는 다음에 따라 coded_sub_block_flag 의 이전 값들을 이용하여 유도된다.

[0276] - 수평 및 수직 서브-블록 위치들 xS 및 yS 는 (xC >> 2) 및 (yC >> 2)로 각각 동일하게 설정된다.

[0277] - 변수 prevCsbf 는 0 으로 설정된다.

[0278] - xS가 (1 << (log2TrafoSize - 2)) - 1 보다 작을 때, 다음을 적용한다.

[0279] prevCsbf += coded_sub_block_flag[xS + 1][yS]

[0280] - yS가 (1 << (log2TrafoSize - 2)) - 1 보다 작을 때, 다음을 적용한다.

[0281] prevCsbf += (coded_sub_block_flag[xS][yS + 1] << 1)

[0282] - 내부 서브-블록 위치들 xP 및 yP 는 (xC & 3) 및 (yC & 3)과 각각 동일하게 설정된다.

[0283] - 변수 sigCtx 는 다음에 따라 유도된다.

[0284] - 만약 prevCsbf 가 0과 같다면, 다음을 적용한다.

[0285] sigCtx = (xP + yP = = 0) ? 2 : (xP + yP < 3) ? 1: 0

[0286] - 그렇지 않다면, 만약 prevCsbf 가 1과 같다면, 다음을 적용한다.

[0287] sigCtx = (yP = = 0) ? 2 : (yP = = 1) ? 1: 0

[0288] - 그렇지 않다면, 만약 prevCsbf 가 2와 같다면, 다음을 적용한다.

[0289] sigCtx = (xP = = 0) ? 2 : (xP = = 1) ? 1: 0

[0290] - 그렇지 않고 (prevCsbf이 3과 같다면), 다음을 적용한다.

[0291] sigCtx = 2

[0292] - 변수 sigCtx 는 다음에 따라 수정된다.

[0293] - 만약 cIdx가 0과 같다면, 다음을 적용한다.

[0294] - (xS + yS) 가 0 보다 크다면, 다음을 적용한다.

[0295] sigCtx += 3

[0296] - 변수 sigCtx 가 다음에 따라 수정된다.

[0297] - 만약 log2TrafoSize 가 3과 같다면, 다음을 적용한다.

[0298] sigCtx += (scanIdx = = 0) ? 9 : 15

[0299] - 그렇지 않다면, 다음을 적용한다.

[0300] sigCtx += 21

[0301] - 그렇지 않고 (cIdx 는 0보다 크다면), 다음을 적용한다.

[0302] - log2TrafoSize 이 3과 동일하다면, 다음을 적용한다.

[0303] sigCtx += 9

[0304] - 그렇지 않다면, 다음을 적용한다.

[0305] sigCtx += 12

[0307] 컨텍스트 인덱스 증가 ctxIdxInc는 다음에 따라 컬러 구성요소 인덱스 cIdx 및 sigCtx를 이용하여 유도된다.

[0308] - cIdx가 0과 같다면, ctxIdxInc 는 다음에 따라 유도된다.

[0309] ctxIdxInc = sigCtx

[0310] - 그렇지 않다면 (cIdx는 0보다 크다), ctxIdxInc는 다음에 따라 유도된다.

[0311] ctxIdxInc = 27 + sigCtx

표 1

[0313] ctxIdxMap [i] 의 사양(specification)

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
ctxIdxMap[i]	0	1	4	5	2	3	4	5	6	6	8	8	7	7	8

[0314] 위에서 설명된 것처럼, 각 중요 변환 계수에 대해, 추가 구문 요소들 또는 기호들의 집합들은 그것들의 레벨을 시그널링하기 위해 데이터 스트림 내에서 운반될 수 있다. 아래에서 설명되는 실시예에 따라, 하나의 중요 변환 계수에 대해 다음 구문 요소들 또는 변환 계수들의 집합들이 전송된다 : coeff_abs_level_greater1_flag, coeff_abs_level_greater2_flag (선택적), and coeff_abs_level_remaining 그래서 현재 코딩된/디코딩된 중요 변환 계수 레벨 TransCoeffLevel 의 레벨은

[0315] TransCoeffLevel = (coeff_abs_level_remaining + baseLevel) * (1 - 2 * coeff_sign_flag]) 이고

[0316] 이와 함께

[0317] baseLevel = 1 + coeff_abs_level_greater1_flag + coeff_abs_level_greater2_flag 이다.

[0319] significant_coeff_flag 는, 정의 당(per definitnion), 중요 변환 계수들에 대해, 따라서, 변환 계수의 코딩의 부분으로 간주될 수 있고, 즉 즉 그것의 엔트로피 코딩된 기호들의 부분들이라는 점에 주목하자.

[0321] 컨텍스트 적응 엔트로피 디코더/인코더는, 예를 들어, 다음에 따라 coeff_abs_level_greater1_flag 에 대한 컨텍스트 선택을 수행할 것이다. 예를 들어, 현재 서브-블록 스캔 인덱스 *i*는 DC의 방향으로 스캔 패스(202)를 따라 증가할 것이고, 현재 계수 스캔 인덱스 n 은 스캔 패스(scan path)(204)를 따라, 현재 코딩된/디코딩된 변환 계수 위치가 위치되는 곳 내의 각 서브-블록 내에서 증가할 것이며, 여기서 위에서 설명되듯이, 스캔 패스(202 및 204)에 대해 상이한 개연성들이 존재하고, 동일한 것이 인덱스 scanIdx에 따라 실제적으로 가변적일 수 있다.

[0323] coeff_abs_level_greater1_flag 의 컨텍스트 선택의 프로세스에 대한 입력들은 컬러 구성요소 인덱스 cIdx, 현재 서브-블록 내의 현재 계수 스캔 인덱스 n 및 현재 서브-블록 스캔 인덱스 i 이다.

[0324] 이 프로세스의 출력은 ctxIdxInc 이다.

[0325] 가변 ctxSet 은 그것의 유도에 대한 그리고 현재 컨텍스트를 특정하고 다음을 적용한다.

[0326] - 만약 이 프로세스는 현재 서브-블록 스캔 인덱스 i에 대한 첫번째에 원용되며, 다음을 적용한다.

[0327] - 가변 ctxSet는 다음에 따라 초기화된다.

[0328] - 만약 현재 서브-블록 스캔 인덱스 i 는 0 과 같고 cIdx는 0보다 크고, 다음을 적용한다.

[0329] ctxSet = 0

[0330] - 그렇지 않다면 (i 는 0보다 크고 cIdx 는 0과 같다), 다음을 적용한다.

[0331] ctxSet = 2

[0332] - 가변 lastGreater1Ctx 는 다음에 따라 유도된다.

[0333] - 만약 스캔 인덱스 i를 갖는 현재 서브-블록이 현재 변환 블록에 대한 서브클로스(하위조항, subclause)에서 처리될 첫번째 것이라면, 가변 lastGreater1Ctx 이 1과 동일하게 설정된다.

[0334] - 그렇지 않다면, 가변 lastGreater1Ctx 는 스캔 인덱스 i + 1를 갖는 이전 서브-블록에 대한 구문 요소

coeff_abs_level_greater1_flag 에 대한 이 하위조항에서 특정되는 프로세스의 최종 호출(invocation) 동안 유도되는 greater1Ctx 의 값에 동일하게 설정된다.

[0335] - lastGreater1Ctx 가 0과 동일할 때, xtcSet은 하나씩 점차 증가(점증)된다.

[0336] ctxSet = ctxSet + 1

[0337] - 가변 greater1Ctx 는 1로 설정된다.

[0338] - 그렇지 않으면 (이 프로세스는 현재 서브-블록 스캔 인덱스 i에 대한 첫번째 것에 대해 호출되지 않는다), 다음을 적용한다.

[0339] - 가변 ctxSet은 이 하위조항에서 특정되는 프로세스의 최종 호출 동안 유도되는 가변 ctxSet 에 동일하게 설정된다.

[0340] - 가변 greater1Ctx 는 이 하위조항에서 특정되는 프로세스의 최종 호출 동안 유도되는 가변 greater1ctx에 동일하게 설정된다.

[0341] - greater1Ctx 는 0보다 크고, 가변 lastGreater1Flag 는 이 하위조항에서 특정되는 프로세스의 최종 호출 동안 이용되는 구문 요소 coeff_abs_level_greater1_flag 에 동일하게 설정되고 greater1Ctx는 다음에 따라 수정된다.

[0342] - lastGreater1Flag 는 1과 동일하게 설정된다면, greater1Ctx는 0으로 설정된다.

[0343] - 그렇지 않으면 (lastGreater1Flag 는 0과 동일하다), greater1Ctx는 1씩 점증한다.

[0346] *컨텍스트 인덱스 증가 ctxIdxInc 는 다음에 따라 현재 컨텍스트 greater1Ctx 및 현재 컨텍스트 집합 ctxSet 를 이용하여 유도된다.

[0347] ctxIdxInc = (ctxSet * 4) + Min(3, greater1Ctx)

[0348] cIdx 가 0보다 클 때, crxIdxInc 는 다음에 따라 수정된다.

[0349] ctxIdxInc = ctxIdxInc + 16

[0351] coeff_abs_level_greater2_flag 의 컨텍스트를 선택하는 프로세스는 다음 차이를 가지고 coeff_abs_level_greater2_flag 처럼 동일하게 만들어질 수 있다. 컨텍스트 인덱스 증가 ctxIdxInc는 다음에 따라 가변 ctxSet 와 동일하게 설정된다.

[0352] ctxIdxInc = ctxSet

[0353] cIdx 는 0보다 클 때, ctxIdxInc는 다음에 따라 수정된다.

[0354] ctxIdxInc = ctxIdxInc + 4

[0356] 기호화 파라미터 선택에 대해, 여기서 cLastAbsLevel 및 cLastRiceParam 를 포함하는 포함하는 기호화 파라미터를 결정하도록 기호화 파라미터 결정기에 의해 다음이 수행된다.

[0358] 이 프로세스에 대한 입력은 구문 요소 coeff_abs_level_remaining[n], 및 baseLavel 에 대해 이진화를 요청한다.

[0359] 이 프로세스의 출력은 구문 요소의 이진화이다.

[0360] 가변 cLastAbsLevel 및 cLastRiceParam 는 다음에 따라 유도된다.

[0361] - n이 15와 동일한 경우, cLastAbsLevel 및 cLastRiceParam 는 0으로 설정된다.

[0362] - 그렇지 않으면 (n 은 15보다 작고), cLastAbsLevel 는 baseLevel + coeff_abs_level_remaining[n + 1] 와 동일하게 설정되고 cLastRiceParam 는 동일 변환 블록의 구문 요소 coeff_abs_level_remaining[n + 1] 에 대한 이 하위조항에서 특정되는 것처럼 이진화 프로세스의 호출 동안 유도되는 cRiceParam의 값과 동일하게 설정된다.

[0364] 가변 cRiceParam 은 cLastAbsLevel 및 cLastRiceParam 으로부터 유도된다:

[0365] cRiceParam =

[0366] Min(cLastRiceParam + (cLastAbsLevel > (3 * (1 << cLastRiceParam)) ? 1 : 0), 4)

[0367] 가변 cTRMax 는 cRiceParam 으로부터 유도된다:

[0368] cTRMax = 4 << cRiceParam

[0369] coeff_abs_level_remaining 의 이진화는 (존재할 때) 접미사 부분 및 접두사 부분으로 구성될 수 있다.

[0370] 이진화의 접두사 부분은, 예를 들어, 접두사 부분 Min (cTRMax, coeff_abs_level_remaining[n]) 에 대한 라이스 이진화 프로세스를 호출하는 것에 의해 유도된다.

[0371] 접두사 빈 스트링(prefix bin string)이 길이 4의 비트 스트링(bit string)에 동일할 때, 예를 들어, 1과 동일한 모든 비트들을 가지며, 빈 스트링은 접두사 빈 스트링 및 접미사 빈 스트링으로 구성될 수 있다. 접미사 빈 스트링은 예를 들어, cRiceParam + 1 와 동일하게 설정되는 Exp-Golomb order k 를 갖는 접미사 부분 (coeff_abs_level_remaining[n] - cTRMax)에 대해 Exp Golomb order-k 이진화를 이용하여 유도된다.

[0373] 위 실시예들이 변화될 수 있다는 것이 주목되어야 한다. 예를 들어, 컬러 구성요소 인덱스 cIdx 에 대한 의존도가 없어진다. 단지 하나의 컬러 구성요소 유드(color component yould)는, 예를 들어, 고려될 수 있다. 게다가, 모든 이용 값들은 변화될 수 있다. 그러는 한, 방금-설명된 예들은 변화들을 포함시키기 위해 넓게 해석될 것이다.

[0375] 위 예들에서, 위에서 설명된 실시예들은 다음 방식으로 유리하게 이용될 수 있다. 특히, 한편으로는 coeff_abs_level_greater1_flag 에 대한 CtxIdxInc 의 결정 및 coeff_abs_level_remaining 에 대한 기호화 파라미터 결정은 다음 방식으로 함수 파라미터들들을 설정하는 것에 의해 위 함수 f 및 g 를 활용하여 조화된다.

[0377] 이런 이유로, 도 16은 예시적으로 십자가 (206)로 도시되는 "현재 변환 계수"를 보여준다. 동일한 것은 이후에 언급된 구문 요소들이 관련되는 것들 중 어느 것을 갖는 어떤 변환 계수에 대한 표현이다. 그것은 현재 서브-블록 (xS,yS)=(0,1) 내에서 (xP,yP)=(1,1) 및 (xC,yC)=(1,5)에서 위치된다. 오른쪽-인근 서브-블록은 (xS,yS)=(1,1)이며, 아래-인근 서브-블록은 (xS,yS)=(0,2)이며 바로 이전 코딩된 서브-블록은 스캔 패스 (202)에 의존한다. 여기서, 예시적으로, 대각선 스캔(202)가 보여지며, 서브-블록 코딩된/디코딩된 바로 선행하는 현재 서브-블록은 (xS,yS)=(1,0)이다.

[0379] 다시, 공통 파라미터화 가능 함수에 대해 공식을 다시 써보자(rewrite).

[0381] $$g(f(x)) = \sum_{i=1}^{d_f} \delta'(f(x), n_i) g(f(x)) = \sum_{i=1}^{d_f} \delta'(f(x), n_i) \quad (1)$$

[0382] $$f(x) = \sum_i w_i \times h(x_i) \times \delta(x_i, t) f(x) = \sum_i w_i \times h(x_i) \times \delta(x_i, t) \quad (2)$$

[0384] 현재 계수(206)에 대한 significant_coeff_flag 의 컨텍스트를 선택하는 것에 대해, 다음이 엔트로피 인코딩/디코딩 장치에 의해 계산될 수 있다. 그것은, 동일한 것이 다음에 따라 함수 파라미터 t, h 및 w 집합을 갖는 함수 (1) 과 (2) 를 이용할 것이다:

[0385] 함수 (2)에 대해, $w_i = 1$는 *현재 서브-블록의 오른쪽* 및 아래쪽에 인접한 서브-블록들 내에서 모든 x_i 에 대해서이고, 블록(10)에서 그 외 다른 곳에서는 w_i=0이다;

[0386] 현재 서브-블록의 오른쪽에 인접 서브-블록 내의 모든 x_i 에 대해 $h(x_i) = 1$ 이다. 만약 존재한다면, 동일한 것이 서브-블록 스캔(202)에서 이전에 스캔된다;이러한 경우, 하나 이상의 스캔(202)이 이용 가능하며, 모든 것들은, scanIdx에 독립적으로, 오른쪽에 인접한 서브-블록이 현재 서브-블록에 앞서 코딩/디코딩된 그것의 계수들을 가질 수 있다.

[0388] 서브-블록 스캔에서 이전에 스캔된 현재 서브-블록 아래의 인접 서브-블록 내의 모든 x_i에 대해 $h(x_i) = 2^4 + 1$ 이고;

[0389] 그렇지 않으면 $h(x_i) = 0$ 이고;

[0390] $t = 1$ 이다;

[0391] 만약 f 의 값이 0과 같다면, 이는 현재 서브-블록 나크반(Nachbarn) 및 오른쪽에 대한 인접 서브 블록들 중 아무것도 어떠한 중요 변환 계수를 포함하지 않는 경우를 시그널링한다.

[0392] 만약 f 가 1 및 16 사이의 범위라면, 둘 다 포함하여, 이는 오른쪽 인접 서브-블록에서 coded_sub_block_flag 가 1에 동일하다는 사실에 대응한다.

[0393] 만약 f 가 $2^4 + 1$ 의 배수라면 (나머지 없이), 이는 아래쪽 인접 서브-블록 에서 coded_sub_block_flag 가 1과 동일하다는 사실에 대응한다.

[0394] 만약 f의 값이 $2^4 + 1$ 의 배수이고, 나머지가 있다면, 이는 즉 현재 블록의 아래 중 하나, 오른쪽에 대한 하나인, 양쪽 인접 서브-블록들에 대해 coded_sub_block_flag 가 1과 동일하다는 의미이다.

[0395] 함수 (1)에 대해, n 은 3인 d_f 에 따라 설정된다.

[0396] n=(0, 2^4, m)

[0397] 와 함께

[0398] $$m = \begin{cases} 2^{16} & if \;\; f(x) \le 2^4 \\ f(x) - f(x)\%(2^4+1) & else \end{cases}$$

[0399] 이 방법에 의해, 컨텍스트 인덱스의 가변 구성요소는 이미 코딩된(디코딩된) 계수들에 기반하여 위 함수 파라미터와 함께 g(f)를 이용하여 결정된다.

[0401] coeff_abs_greater1_flag 의 컨텍스트를 선택하는 것에 대해, 다음이 엔트로피 인코딩/디코딩 장치에 의해 계산될 수 있다. 그것은, 다음에 따라 설정되는 함수 파라미터를 갖는 것과 함께 동일한 것이 함수 (1) 과 (2)를 이용할 것이라는 것이다.

[0403] 함수 (2)에 대해, 파라미터들은 다음에 따라 설정된다.

[0404] 모든 다른 것들에 대해 0이고, 현재 서브-블록 및 바로 선행 서브-블록에서 모든 x_i에 대해 $w_i = 1$이다.

[0405] $|x_i| = 1$ 와 함께 현재 서브-블록에서 모든 x_i에 대해 $h(x_i) = 1$ 이다.

[0407] * $|x_i| > 1$ 인 현재 서브-블록에서 모든 x_i에 대해 $h(x_i) = 2^4$ 이다.

[0408] 즉시 선행 서브-블록에서 모든 x_i에 대해 $h(x_i) = 2^{16}$ 이고

[0409] $t = 2$ 이다.

[0410] 함수 (1)에 대해 n은 8인 d_f 와 함께 다음에 따라 설정된다.

[0411] $$n = (0,1,2,2^4,2^{16},2^{16}+1,2^{16}+2,2^{16}+2^4)$$

[0413] coeff_abs_greater2_flag 의 컨텍스트를 선택하는 것에 대해, 다음이 엔트로피 인코딩/디코딩 장치에 의해 계산될 수 있다. 특히, 동일한 것은 coeff_abs_greater2_flag 에 관해 위에서 설명된 것처럼 설정된 함수 파라미터를 갖고 함수 (1) 과 (2) 를 이용할 수 있고, 다만 1인 d_f 와 함께이다.

[0415] $$n = (2^{16})$$

[0417] coeff_abs_level_remaining 에 대한 기호화 파라미터를 결정하기 위해, 기호화 파라미터 결정기는 다음에 따라 함수 파라미터들과 함께 공통 함수 (1)를 이용할 수 있다.

[0418] 함수 (2)에 대해, 파라미터들은 다음에 따라 설정된다.

[0419] 현재 서브-블록에서 모든 x_i에 대해 $w_i = 1$이나, 다른 곳에서는 0이다.

[0420] coeff_abs_level_remaining 이 코딩되는 - 내부 계수 스캔 (204)에 따라 - 가장 최근에 방문된 계수 x_i에 대해 $h(x_i) = 1$이며, 즉 즉, 기호화 설계에 따르는 인터벌의 범위에 드는 레벨이다;

[0421] 템플릿의 다른 곳에서 $h(x_i) = 0$이고

[0422] $t = 0$이다

[0424] 함수 (1)에 대해 n 은 다음에 따라 설정된다.

[0425] $m = \begin{cases} k \ if k < 4 \\ 2^{16} if k = 4 \end{cases}$와 함께 $\boldsymbol{n} = (2^m)$ 이며 여기서 k는 -내부 계수 스캔(204)에 따라- 앞서 언급된 가장 최근에 방문된 계수에 대한, 기호화 파라미터이고, 예를 들어, 라이스 파라미터이다. 결과 g(f)를 이용하여, 현재 계수 (206)에 대한 기호화 파라미터가 결정된다.

[0427] 다음 구문은 방금-설명된 구문 요소들을 전송하는데 이용될 수 있다.

[0429]
```
residual_coding( x0, y0, log2TrafoSize, cIdx ) {
```

[0430]
```
    if( transform_skip_enabled_flag && !cu_transquant_bypass_flag && ( log2TrafoSize  = =  2 ) )
```

[0431]
```
        transform_skip_flag[ x0 ][ y0 ][ cIdx ]
```

[0432]
```
    last_significant_coeff_x_prefix
```

[0433]
```
    last_significant_coeff_y_prefix
```

[0434]
```
    if( last_significant_coeff_x_prefix > 3 )
```

[0435]
```
        last_significant_coeff_x_suffix
```

[0436]
```
    if( last_significant_coeff_y_prefix > 3 )
```

[0437]
```
        last_significant_coeff_y_suffix
```

[0438]
```
    lastScanPos = 16
```

[0439]
```
    lastSubBlock = ( 1 << ( log2TrafoSize - 2 ) ) * ( 1 << ( log2TrafoSize - 2 ) ) - 1
```

[0440]
```
    do {
```

[0441]
```
        if( lastScanPos  = =  0 ) {
```

[0442]
```
            lastScanPos = 16
```

[0443]
```
            lastSubBlock- -
```

[0444]
```
        }
```

[0445]
```
        lastScanPos- -
```

[0446]
```
        xS = ScanOrder[ log2TrafoSize - 2 ][ scanIdx ][ lastSubBlock ][ 0 ]
```

[0447]
```
        yS = ScanOrder[ log2TrafoSize - 2 ][ scanIdx ][ lastSubBlock ][ 1 ]
```

[0448]
```
        xC = ( xS << 2 ) + ScanOrder[ 2 ][ scanIdx ][ lastScanPos ][ 0 ]
```

[0449]
```
        yC = ( yS << 2 ) + ScanOrder[ 2 ][ scanIdx ][ lastScanPos ][ 1 ]
```

[0450]
```
    } while( ( xC  !=  LastSignificantCoeffX ) | | ( yC  !=  LastSignificantCoeffY ) )
```

```
[0451]          for( i = lastSubBlock; i >= 0; i- - ) {
[0452]                  xS = ScanOrder[ log2TrafoSize - 2 ][ scanIdx ][ i ][ 0 ]
[0453]                  yS = ScanOrder[ log2TrafoSize - 2 ][ scanIdx ][ i ][ 1 ]
[0454]                  inferSbDcSigCoeffFlag = 0
[0455]                  if( ( i < lastSubBlock )  && ( i > 0 ) ) {
[0456]                          coded_sub_block_flag[ xS ][ yS ]
[0457]                          inferSbDcSigCoeffFlag = 1
[0458]                  }
[0459]                  for( n = ( i  = =  lastSubBlock ) ? lastScanPos - 1 : 15; n >= 0; n- - ) {
[0460]                          xC = ( xS << 2 ) + ScanOrder[ 2 ][ scanIdx ][ n ][ 0 ]
[0461]                          yC = ( yS << 2 ) + ScanOrder[ 2 ][ scanIdx ][ n ][ 1 ]
[0462]                          if( coded_sub_block_flag[ xS ][ yS ]  &&  ( n > 0  | |  !inferSbDcSigCoeffFlag
      ) ) {
[0463]                                  significant_coeff_flag[ xC ][ yC ]
[0464]                                  if( significant_coeff_flag[ xC ][ yC ] )
[0465]                                          inferSbDcSigCoeffFlag = 0
[0466]                          }
[0467]                  }
[0468]                  firstSigScanPos = 16
[0469]                  lastSigScanPos = -1
[0470]                  numGreater1Flag = 0
[0471]                  lastGreater1ScanPos = -1
[0472]                  for( n = 15; n >= 0; n- - ) {
[0473]                          xC = ( xS << 2 ) + ScanOrder[ 2 ][ scanIdx ][ n ][ 0 ]
[0474]                          yC = ( yS << 2 ) + ScanOrder[ 2 ][ scanIdx ][ n ][ 1 ]
[0475]                          if( significant_coeff_flag[ xC ][ yC ] ) {
[0476]                                  if( numGreater1Flag < 8 ) {
[0477]                                          coeff_abs_level_greater1_flag[ n ]
[0478]                                          numGreater1Flag++
[0479]                                          if( coeff_abs_level_greater1_flag[ n ] && lastGreater1ScanPos
      = =  -1 )
[0480]                                                  lastGreater1ScanPos = n
[0481]                                  }
[0482]                                  if( lastSigScanPos  = =  -1)
[0483]                                          lastSigScanPos = n
[0484]                                  firstSigScanPos = n
```

[0485] }

[0486] }

[0487] signHidden = (lastSigScanPos - firstSigScanPos > 3 && !cu_transquant_bypass_flag)

[0488] if(lastGreater1ScanPos != ?1)

[0489] coeff_abs_level_greater2_flag[lastGreater1ScanPos]

[0490] for(n = 15; n >= 0; n- -) {

[0491] xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]

[0492] yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]

[0493] if(significant_coeff_flag[xC][yC] &&

[0494] (!sign_data_hiding_flag | | !signHidden | | n != firstSigScanPos))

[0495] coeff_sign_flag[n]

[0496] }

[0497] numSigCoeff = 0

[0498] sumAbsLevel = 0

[0500] * for(n = 15; n >= 0; n- -) {

[0501] xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]

[0502] yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]

[0503] if(significant_coeff_flag[xC][yC]) {

[0504] baseLevel = 1 + coeff_abs_level_greater1_flag[n] + coeff_abs_level_greater2_flag[n]

[0505] if(baseLevel = = ((numSigCoeff < 8) ? ((n = = lastGreater1ScanPos) ? 3 : 2) : 1))

[0506] coeff_abs_level_remaining[n]

[0507] TransCoeffLevel[x0][y0][cIdx][xC][yC] =

[0508] (coeff_abs_level_remaining[n] + baseLevel) * (1 - 2 * coeff_sign_flag[n])

[0509] if(sign_data_hiding_flag && signHidden) {

[0510] sumAbsLevel += (coeff_abs_level_remaining[n] + baseLevel)

[0511] if(n = = firstSigScanPos && ((sumAbsLevel % 2) = = 1))

[0512] TransCoeffLevel[x0][y0][cIdx][xC][yC] = -TransCoeffLevel[x0][y0][cIdx][xC][yC]

[0513] }

[0514] numSigCoeff++

[0515] }

[0516] }

[0517] }

[0518] }

[0520] 상기 구문은 변환 계수의 레벨이 *baseLevel* 및 *coeff_abs_level_remaining* 로 구성된다는 것을 나타내며, 여기서 *baseLevel* 은 1 + *coeff_abs_level_greater1_flag[n]* + *coeff_abs_level_greater2_flag[n]* 로 구성된다. 1이 이용되며, 이러한 위치에서처럼(또는 레벨들이 디코더에서 복원되는 시간에서) 구문 요소는 significant_coeff_flag=1 이다.

[0522] "제1집합(첫번째 집합, first set)"은 TU 코드 (0과 동일한 파라미터화를 갖는 라이스 코드) - 이로부터 첫번째 3 구문 요소들이 형성된다. "제2집합(두번째 집합, second set)"은 이후 구문 요소 *coeff_abs_level_remaining* 를 형성한다.

[0524] 경계가 "첫번째" 및 "두번째 집합" 사이에서 이동되기 때문에(시프팅 되기 ㄸ때맴때문에, shifted), 최대 값은 *coeff_abs_greater1_flag*, *coeff_abs_greater2_flag* 또는 *significant_coeff_flag* 에 의해 각각 정의되고, 이런 이유로 상기 브랜치(branches)는 표의 구문 요소들에 의존한다.

[0526] 함수 파라미터의 위 설정들은 다음에서 조금 여전히 동기가 부여된다. g(f) 는 인접 계수들의 합을 형성하고 결과 컨텍스트를 이용하며 역기호화 파라미터가 유도되며, 여기서 나중의 수정은 공간적 위치에 의존하여 실행될 수 있다.

[0528] g(x) 는 하나의 단일 값을 획득한다. 이 값은 함수 f(x) 의 결과에 대응한다. 이러한 점을 알고, 컨텍스트 선택 및 라이스 파라미터의 파라미터화가 유도될 수 있다.

[0530] **significant_coeff_flag**: h 가 x의 함수 그 자체일 수 있기 때문에, f(x) 또는 어떠한 다른 함수는 계속해서 연계(chained)될 수 있다. 함수 f(x)는 오른쪽 4x4 서브-블록에서 모든 위치들에 대해 w_i=1를 가지며, h 함수 및 t=1는 f(x) 같이 구성되지만 반전되며(inverted), 그래서 마지막에서 값 0 또는 1은 즉, h(x)=min(1, f(x)) 를 도출한다.

[0532] 균등하게, 제2입력에 대해 이는 아래쪽 4x4 서브-블록에 적용된다. 이후, prevCsbf=h_0+2×h_1이고, 여기서 prefCsbf는 f(x) 내에서 함수 h 일 수도 있다.

[0534] t=∞로 설정되는 경우, 구문 요소 *coded_sub_block_flag* 의 값이 유도될 수 있다. g(x) 에 대한 파라미터 n은 (xP+yP), xP, yP, 또는 (0, 0) 중 하나일 것이다. f(x)=0을 도출하는 경우, f(x)=1에 대해 n=(yP,yP+1)가 도출되고, f(x)=2에 대해 n=(xP+yP,xP+yP+3) 이 도출되고, f(x)=3 에 대해 n=(0,0)이 도출된다. 말하자면, f(x)는 n을 결정하기 위해 바로 측정될 수 있다. 위의 나머지 공식은 단순히 전체적 위치 및 스캔에 대한 추가 의존도 및 루마/채도 상에 의존하는 적응을 설명한다. 순수한 4x4 블록의 경우, f(x) 는 prevCsbf=4 에 대한 값이 (다를 수도 있고) 맵핑 표는 재현될 수 있도록 구성될 수 있다.

[0536] **coeff_abs_level_greater1_flag**: 여기서, 서브-블록들의 측정은 유사하며, 오직 선행 서브-블록만이 측정된다. 결과는, 예를 들어 1 또는 2이며 (그것은 오직 두개의 다른 값들이어야 한다), t=2이다. 이는 선행 서브-블록에서 이미 디코딩된 레벨들에 의존하여 베이스 인덱스의 선택에 대응한다. 서브-블록 내에 위치되는 레벨들 상의 직접 의존도가 획득될 수 있다. 효율적으로, (1로부터 시작되어 3에 제한되는) 0이 디코딩되었을 때 하나의 인덱스에 의해 스위칭이 실행되며 1이 디코딩되자마자 그것은 0으로 설정된다. 상기 배치가 고려되지 않는다면, 파라미터화는 0으로부터 시작하여, 다음에 따라 실행된다. t=3이고 동일 서브-블록에서 모든 레벨들에 대해 w_i=1 이며, 즉 f(x)는 coeff_abs_greater1_flag=1 와 함께 레벨들의 숫자를 제공한다. 추가 함수 f(x)에 대해 t=2이고, 즉, 인코딩된 구문 요소 coeff_abs_greater1_flag 와 함께 위치들의 숫자이다. 첫번째 함수는 제한되며, 즉 h_0=f(x)=min(f_0(x),2) 이고 두번째 함수는 h_1=f(x)=max(f_1(x),1)로 제한된다. 이 연계된 델타 함수(delta function) 모두는 (h_1=1 이면 0이고, 그렇지 않으면 h_0 이다) coeff_abs_greater2_flag 에 대해 오직 집합의 유도가 이용된다 (w_i 는 연계된 내부 함수에 대해 0으로 설정된다).

[0538] **coeff_abs_level_remaining**: 상기 선택은 위에서 설명된대로 현재 서브-블록 및 n에만 제한된다.

[0540] 방금 설명된 실시예에 관해, 다음이 주목된다. 특히, 위 설명과 일치되게, 상이한 개연성들이 템플릿의 정의와 관련하여 존재한다: 상기 템플릿은 무빙 템플릿(moving template)일 수 있고, 그 위치는 현재 계수(206)의 위치 상에 의존하여 결정될 수 있다. 그러한 예시적 무빙 템플릿의 설명은 도 13에서 파선(dashed line, 208)로 설명된다. 상기 템플릿은, 현재 계수(206)가 위치되는, 현재 서브-블록, 현재 서브-블록 아래 및 오른쪽에 대한 인

접 서브-블록들, 뿐만 아니라 위에서 설명되는 대로 스캔 인덱스를 이용하여 하나가 선택가능한 것들 중 몇몇이 있는 경우 서브-블록 스캔(202) 중 어느 것 또는 서브-블록 스캔(202)에서 현재 서브-블록을 즉각 선행하는 하나 이상의 서브-블록들로 구성된다. 대안적으로, 템플릿(208)은 블록(10)의 모든 변환 계수들(12)를 단순히 포함할 수 있다.

[0542] 위 예에서, h 및 n의 값을 선택하기 위한 추가로 상이한 개연성들이 있다. 이러한 값들은, 따라서, 다르게 설정될 수 있다. 1로 설정되는 그러한 가중치(weights)들이 관련되는 한, 이는 어떻게든 w_i 에 관해 참(true)이다. 동일한 것이 또다른 비-제로 값으로 설정될 수 있다. 그것들은 서로 동일해야 할 필요는 없다. w_i 가 $h(x_i)$와 곱해질 때, 동일한 곱셈 값이 상이하게 설정된 비-제로 w_i's 에 의해 달성될 수 있다. 게다가, 기호화 파라미터는 라이스 파라미터가 되어야 할 필요는 없고, 다르게 말하면, 기호화 설계는 라이스 기호화 설계에 제한되지 않는다. 컨텍스트 인덱스 선택에 관해, 예를 들어, 구문 요소의 각 타입에 대해 특정적인, 즉 for significant_coeff_flag, coeff_abs_level_greater1_flag, and coeff_abs_level_greater2_flag 에 대해 특정적인, 몇몇 오프셋 인덱스에 함수 g(f)를 이용하여 얻어질 때 컨텍스트 인덱스를 더하여 최종 컨텍스트 인덱스가 얻어질 수 있다는 것이 이미 알려진 위 설명이 언급된다.

[0544] 비록 몇몇 관점들이 장치의 관점에서 설명되었지만, 이러한 관점들은 또한 대응하는 방법의 묘사도 나타낸다는 것이 명백하며, 여기서 블록 또는 장치는 방법 단계 또는 방법 단계의 특징에 대응한다. 유사하게, 방법 단계의 문맥에서 설명된 관점들은 대응하는 장치의 대응하는 블록 또는 아이템 또는 특징의 설명 또한 나타낸다. 방법 단계들의 몇몇 또는 전부는 하드웨어 장치, 예를 들어, 마이크로프로세서, 프로그램가능한 컴퓨터 또는 전기 회로 등과 같은 것에 의해 (또는 이용하여) 수행될 수 있다. 몇몇 실시예들에서, 가장 중요한 방법 단계들의 몇몇 또는 그 이상은 그러한 장치에 의해 실행될 수 있다.

[0546] 특정한 실행의 요구들에 의존하여, 이 발명의 실시 예들은 하드웨어 또는 소프트웨어에서 실행될 수 있다. 실행들은 전자적으로 읽을 수 있는 컨트롤 신호들을 그곳에 저장하고 있는 디지털 저장매체, 예를 들어 플로피 디스크, DVD, CD, ROM, PROM, EPROM, EEPROM 또는 플래시 메모리,를 이용하여 수행될 수 있고 그것은, 각 방법이 수행되는, 프로그래밍 가능한 컴퓨터 시스템과 연동한다(또는 연동할 수 있다). 그래서, 디지털 저장 매체는 컴퓨터 판독가능할 수 있다.

[0548] 본 발명에 따른 몇몇 실시 예들은 전자적 판독 가능한 컨트롤 신호들을 갖는 데이터 캐리어를 포함하며, 그것은 여기서 설명된 방법 중 하나가 수행되는 프로그래밍 가능한 컴퓨터 시스템과 연동 가능하다.

[0550] 일반적으로 본 발명의 실시 예들은 프로그램 코드로 컴퓨터 프로그램 결과물에서 실행될 수 있으며, 상기 프로그램 코드는 컴퓨터 프로그램 결과물이 컴퓨터에서 수행될 때 상기 방법 중 하나를 수행하도록 작동되는 것이다. 프로그램 코드는 예시적으로 기계 판독가능 캐리어에 저장될 수도 있다. 다른 실시 예들은 여기에 설명되고, 기계 판독가능 캐리어에 저장된 방법들 중 하나를 수행하기 위한 컴퓨터 프로그램을 포함한다.

[0552] 다른 말로, 발명의 방법의 실시 예는, 컴퓨터 프로그램이 컴퓨터에서 운영될 때 여기서 설명된 방법 중 하나를 수행하기 위한 프로그램 코드를 갖는 컴퓨터 프로그램이다.

[0554] 발명의 방법의 또 다른 실시 예는, 여기서 설명된 방법 중 하나를 수행하기 위한 컴퓨터 프로그램을 그 자체에 포함하는 데이터 캐리어이다.(또는 디지털 저장 매체, 또는 컴퓨터 판독가능 매체). 데이터 캐리어, 디지털 저장 매체 또는 저장된 매체는 일반적으로 유형이고 그리고/또는 비-일시적일 수 있다.

[0556] 발명의 방법의 또 다른 실시 예는, 여기서 설명된 방법 중 하나를 수행하기 위한 컴퓨터 프로그램을 나타내는 신호들의 순서 또는 데이터 스트림이다. 데이터 스트림 또는 신호들의 순서는, 예를 들어 인터넷 같은 데이터 통신 연결을 통해 전송되기 위해 예시적으로 구성될 수 있다.

[0558] 또다른 실시 예는 여기서 설명된 방법 중 하나를 수행하기 위해 구성되거나 적응되기 위하여 프로세싱 수단, 예를 들어 컴퓨터 또는 프로그래밍 가능한 논리 장치를 포함한다.

[0560] 또다른 실시 예는 여기서 설명된 방법 중 하나를 수행하기 위한 컴퓨터 프로그램이 그 자체에 설치된 컴퓨터를 포함한다.

[0562] 본 발명에 따른 추가 실시예는 리시버에 대해 여기에 설명된 방법 중 하나를 수행하기 위한 컴퓨터 프로그램을 전송 (예를 들어, 전기적으로 또는 광학적으로) 하도록 구성되는 장치 또는 시스템을 포함한다. 상기 리시버는, 예를 들어, 컴퓨터, 모바일 장치, 메모리 장치 또는 유사품일 수 있다. 상기 장치 또는 시스템은, 예를 들어,

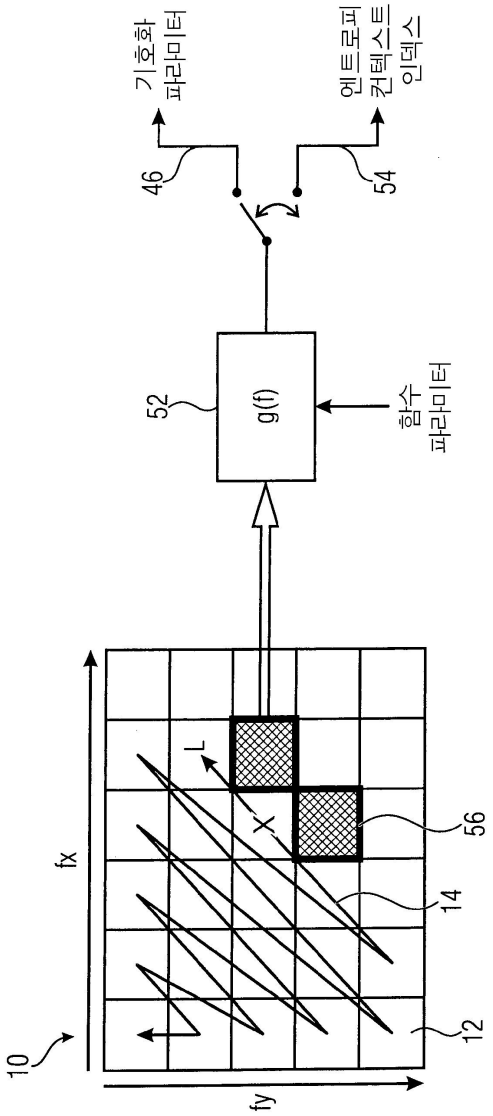
리시버에 컴퓨터 프로그램을 전송하기 위한 파일 서버를 포함할 수 있다.

[0564] 몇몇 실시 예에서, 프로그래밍 가능한 논리 장치(예를 들어 필드 프로그래밍 가능한 게이트 어레이)는 여기서 설명된 방법 중 모든 기능 또는 몇몇을 수행하도록 사용될 수 있다. 몇몇 실시 예에서, 필드 프로그래밍 가능한 게이트 어레이는 여기서 설명된 방법 중 하나를 수행하기 위해 마이크로 프로세서와 연동될 수 있다. 일반적으로, 상기 방법들은 바람직하게는 어떠한 하드웨어 장치에 의해서도 수행된다.

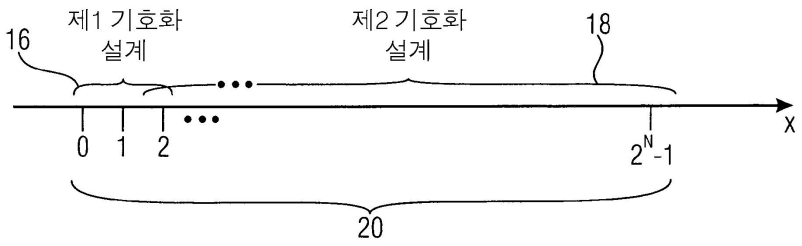
[0566] 상기 설명된 실시 예들은 단지 본 발명의 원리를 위해 예시적일 뿐이다. 본 상기 배열의 변형, 변화, 그리고 여기서 설명된 자세한 내용들을 기술분야의 다른 숙련자에게 명백하다고 이해되어야 한다. 그것의 의도는, 따라서, 여기의 실시 예의 설명 또는 묘사의 방법에 의해 표현된 특정 세부사항들에 의해 제한되는 것이 아닌 오직 목전의 특허 청구항의 범위에 의해서만 제한된다는 것이다.

도면

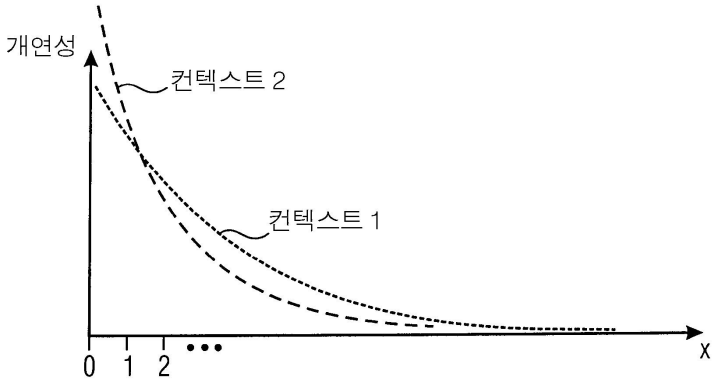
도면1



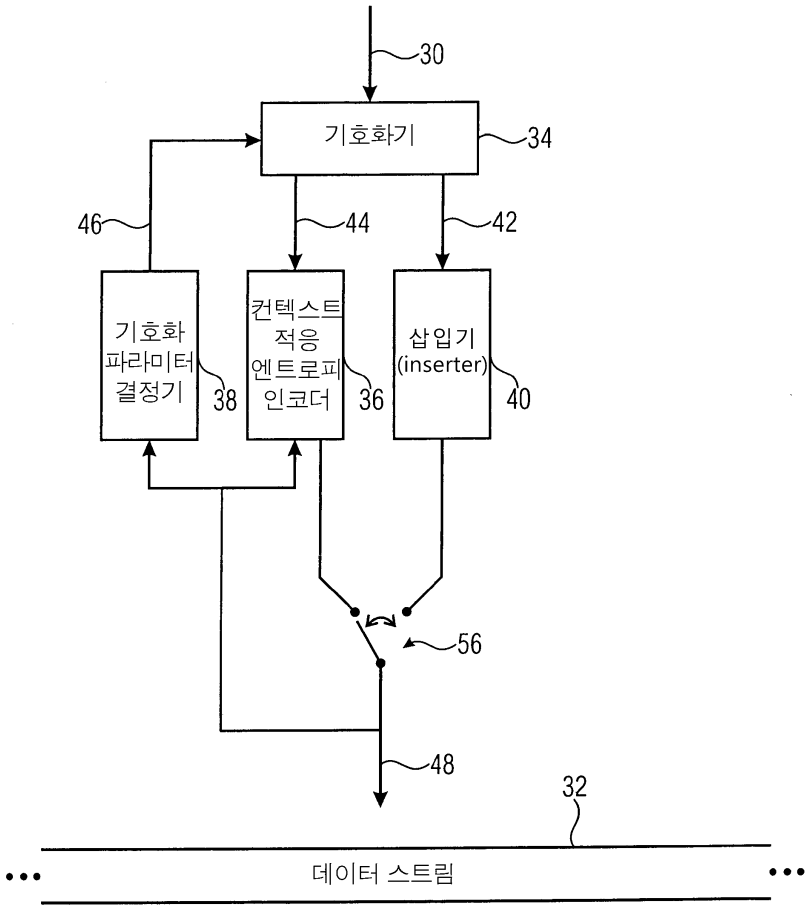
도면2



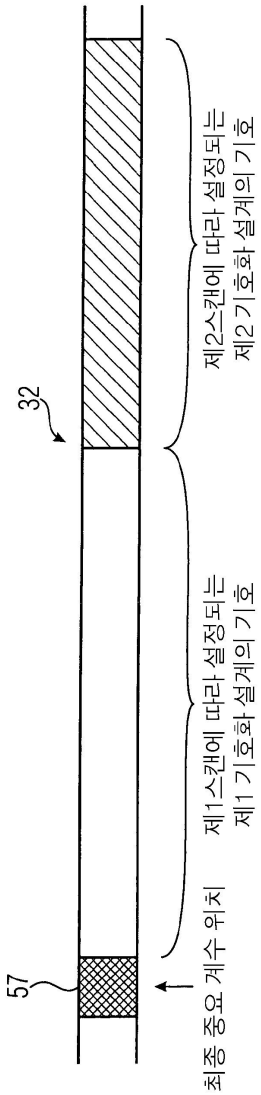
도면3



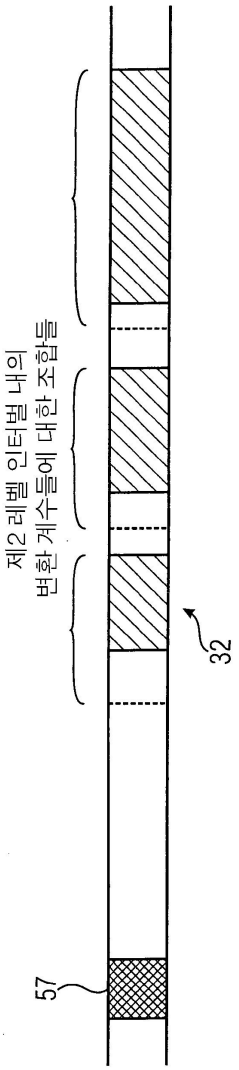
도면4


30
기호화기
34
46
44
42
기호화 파라미터 결정기
38
컨텍스트 적응 엔트로피 인코더
36
삽입기 (inserter)
40
56
48
32
데이터 스트림

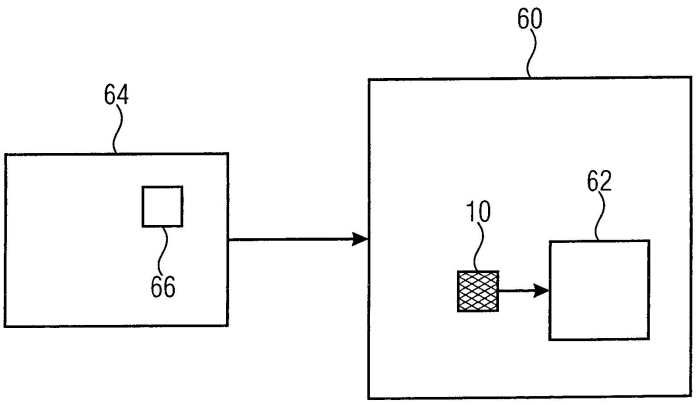
도면5a


32
57
최종 중요 계수 위치
제1스캔에 따라 설정되는
제1 기호화 설계의 기호
제2스캔에 따라 설정되는
제2 기호화 설계의 기호

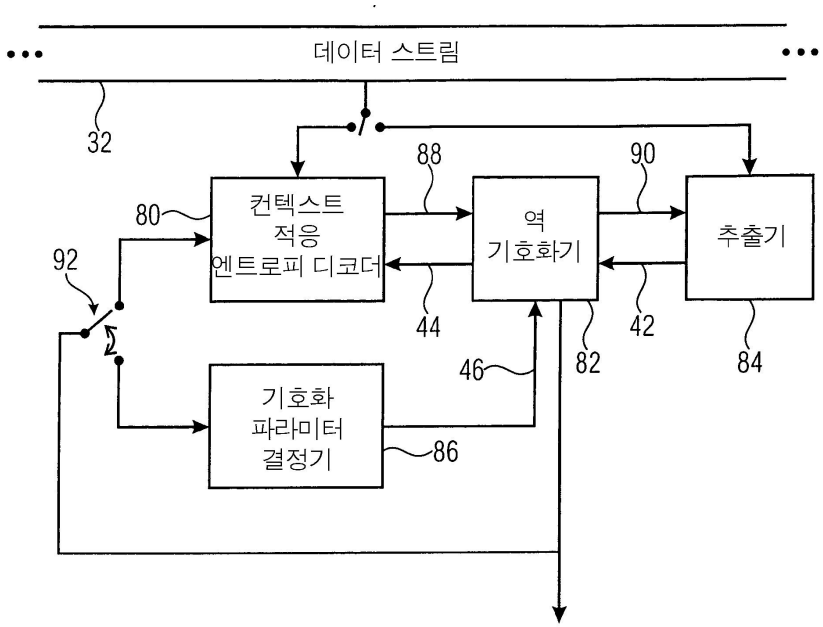
도면5b



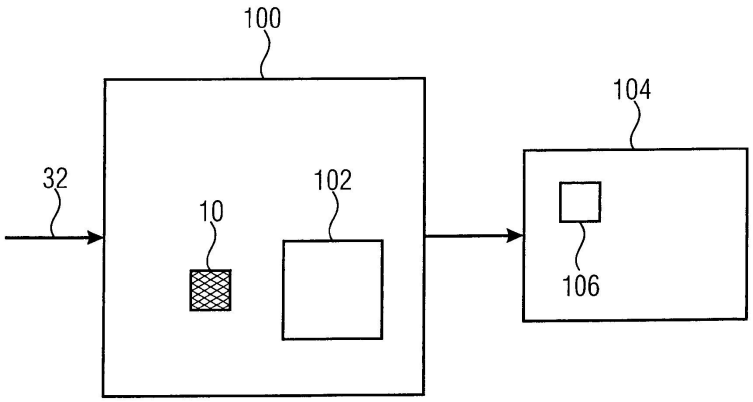
도면6



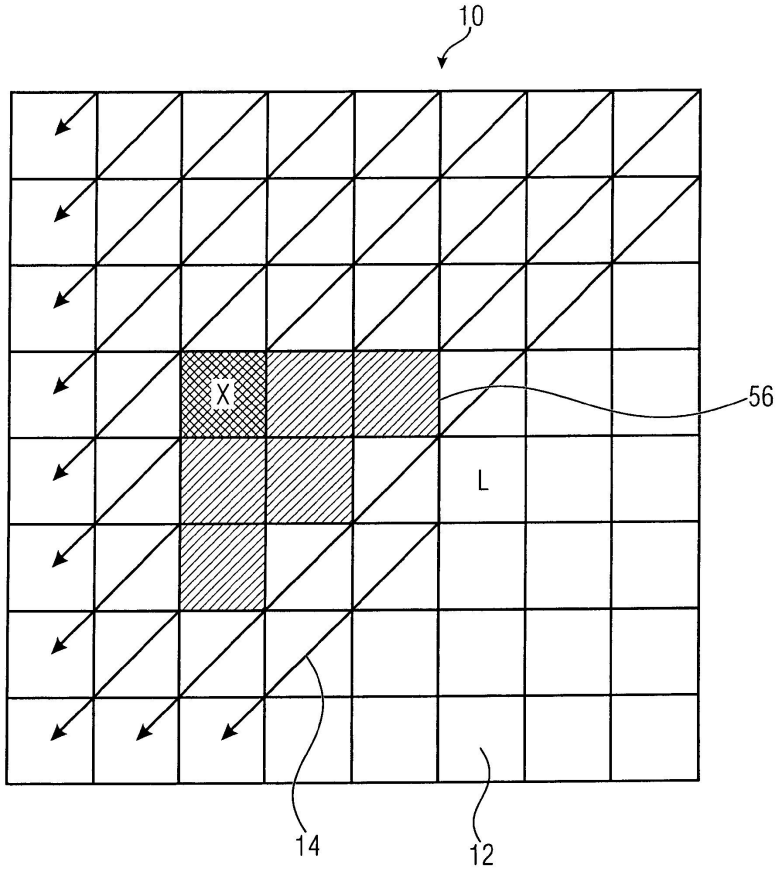
도면7


데이터 스트림
32
88
90
80
컨텍스트
적응
엔트로피 디코더
역
기호화기
추출기
92
44
42
46
82
84
기호화
파라미터
결정기
86

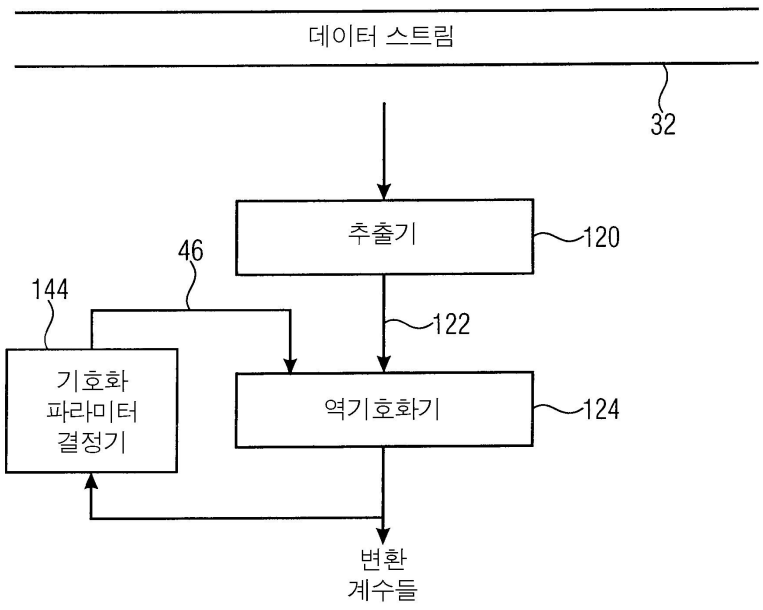
도면8


100
104
32
102
10
106

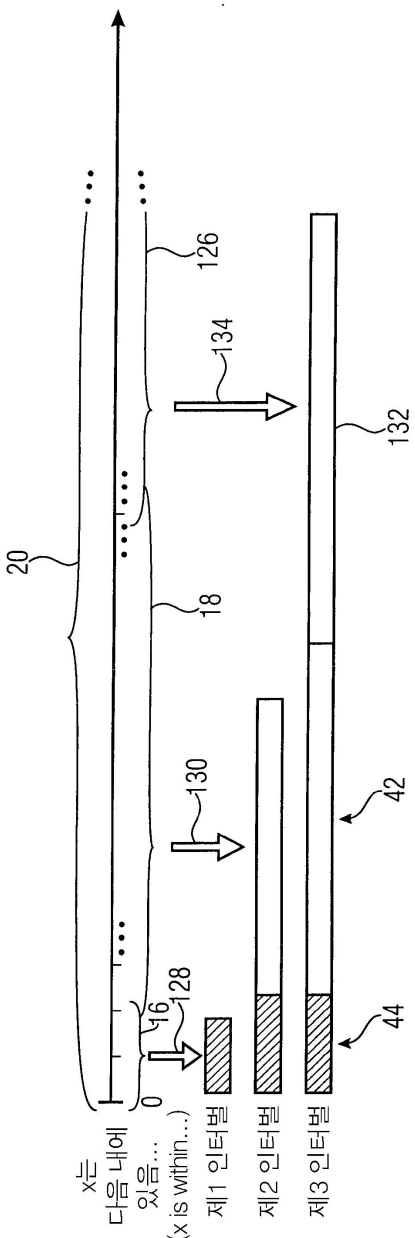
도면9


10
X
56
L
14
12

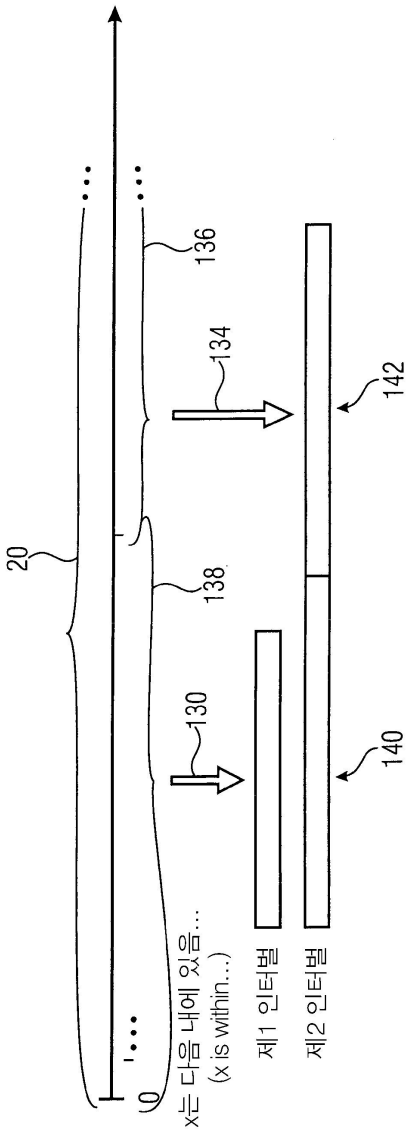
도면10


데이터 스트림
32
추출기
120
46
144
122
기호화
파라미터
결정기
역기호화기
124
변환
계수들

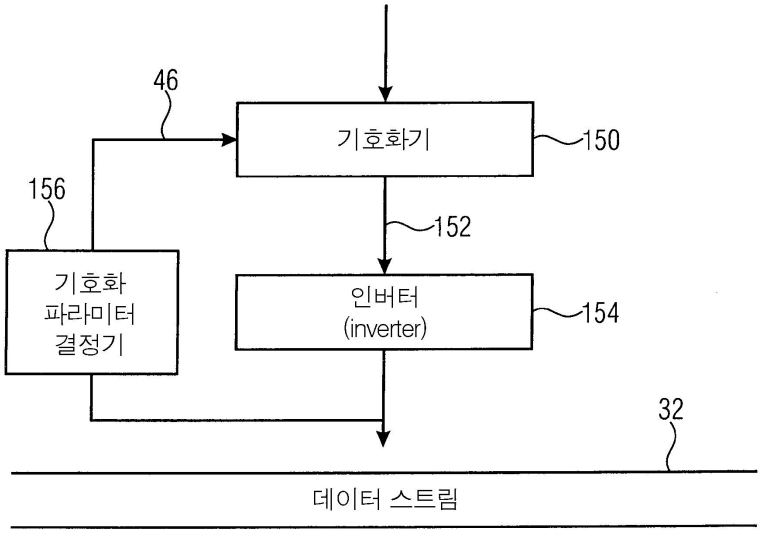
도면11a


20
18
126
134
132
130
42
44
16
128
0
x는
다음 내에
있음...
(x is within...)
제1 인터벌
제2 인터벌
제3 인터벌

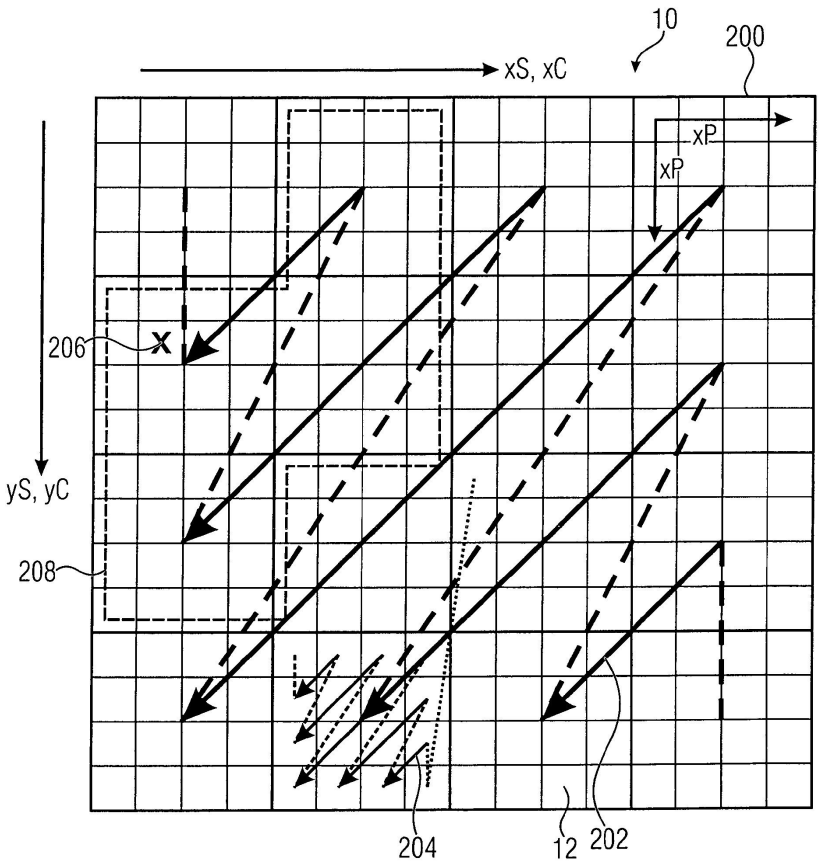
도면11b



도면12


46
기호화기
150
156
152
기호화
파라미터
결정기
인버터
(inverter)
154
32
데이터 스트림

도면13


10
200
xS, xC
xP
xP
206
X
yS, yC
208
204
12
202